Technical Report 8??

Redundant Sensors for Mobile Robot Navigation

AD-A161 087

20000801177

MIT [illegible]

DTIC FILE COPY

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER AI-TR-8[illegible]	2. GOVT ACCESSION NO AD A161037	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) Redundant Sensors for Mobile Robot Navigation		5. TYPE OF REPORT & PERIOD COVERED Technical Report
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Anita M. Flynn		8. CONTRACT OR GRANT NUMBER(s) N00014-80-C-0505
9. PERFORMING ORGANIZATION NAME AND ADDRESS Artificial Inteligence Laboratory 545 Technology Square Cambridge, MA 02139		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS Advanced Research Projects Agency 1400 Wilson Blvd. Arlington, VA 22209		12. REPORT DATE September, 1985
		13. NUMBER OF PAGES 70
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office) Office of Naval Research Information Systems Arlington, VA 22217		15. SECURITY CLASS (of this report) Unclassified
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Distribution is unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

None

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Mobile Robot
Sensors
Map Making
Path Planning
Navigation

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

Redundant sensors are needed on a mobile robot so that the accuracy with which it perceives its surroundings can be increased. Sonar and infrared sensors are used here in tandem, each compensating for deficiencies in the other. The robot combines the data from both sensors to build a representation which is more accurate than if either sensor were used alone. Another representation, the curvature primal sketch, is extracted from

20.

this perceived workspace and is used as the input to two path planning programs: one based on configuration space and one based on a generalized cone formulation of free space.

Redundant Sensors for Mobile Robot Navigation

by

Anita M. Flynn

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

October, 1985

Revised version of a thesis subitted to the Department of Electrical Engineering and Computer Science on August 9, 1985 in partial fulfillment of the requirements for the Degree of Master of Science.

A-1

Table of Contents

Table of Figures

This report describes research done at the Artificial Intelligence Laboratory of the Massachusetts Institute of Technology. Support for the laboratory's artificial intelligence research is provided in part by the Advanced Research Projects Agency of the Department of Defense under Office of Naval Research contract N00014-80-C-0505.

Acknowledgments

I'd like to thank Bart Everett for all the time and help he's given me in working on this thesis. He's a master robot builder and I learned alot from him in the many hours we spent hacking together.

Thanks to Tom McKnight and Russ Werneth of the Naval Surface Weapons Center who provided me with the opportunity, support, and freedom necessary to work on this project.

I would like to thank my advisor, Mike Brady, for good advice on all sorts of things, including a few minor crises. His leaving, which turned out to be the major stimulus in finishing this work, has also caused me much sadness. I'll miss his cheeriness around this lab.

Thanks also to Rod Brooks for getting some of his path planners up and running. Rod was also instrumental in providing me with a great source of thesis procrastination - our own mobile robot at the AI Lab.

I'd like to thank my parents and all my brothers and sisters for their moral support and just plain caring.

Finally, thanks to John Harris for somehow managing to stay my best friend all these years at MIT.

Redundant Sensors for Mobile Robot Navigation

by

Anita M. Flynn

Submitted to the
Department of Electrical Engineering and Computer Science
on August 9, 1985 in partial fulfillment of the requirements
for the Degree of Master of Science

Abstract

Redundant sensors are needed on a mobile robot so that the accuracy with which it perceives its surroundings can be increased. Sonar and infrared sensors are used here in tandem, each compensating for deficiencies in the other. The robot combines the data from both sensors to build a representation which is more accurate than if either sensor were used alone. Another representation, the curvature primal sketch, is extracted from this perceived workspace and is used as the input to two path planning programs: one based on configuration space and one based on a generalized cone formulation of free space.

Academic Supervisor:	Dr. J. Michael Brady Sr. Research Scientist
Company Supervisor:	Dr. Thomas R. McKnight Director Robotics R&D Laboratory Naval Surface Weapons Center

Chapter One

Introduction

In order for a mobile robot to maneuver through its environment and execute any sort of reasonably intelligent task, it should first be able to perceive. That is, it should be able to build a model of its world based on sensory information. An alternative approach might be to assume a near perfect model of the world, and perform task planning from that, as is often done in CAD-based manipulator robotics [Lozano-Perez 81, Brooks 84]. However, in contrast to stationary arm robots which are fixed to a global coordinate frame, a mobile robot's world is essentially unknowable due to cumulative errors, and sensing must be done.

Various types of sensors have been used in the past, such as bumper switches, shaft encoders, sonar transducers [Chattergy 85], photocells and infrared proximity sensors [Everett 82a], cameras [Nilsson 69a, Moravec 81a], infrared beacons [Giralt 83] and laser rangefinders [Thompson 79]. Each type of sensor, however, has some limitation. Shaft encoders aren't accurate when wheels slip, for instance. Sonar sensors have a wide beamwidth and are sensitive to specular surfaces and cameras require computationally intensive processing. One solution, that which is followed here, is to use redundant sensors and utilize the advantageous characteristics of some in overcoming the disadvantages of others.

This thesis describes how two very inexpensive sensors, a sonar rangefinder and a novel infrared sensor, are coupled to produce data that is better for building a representation of the robot's environment than using the sensors individually. The sonar rangefinder measures the distance to an object but has poor angular resolution due to its wide beamwidth. In contrast, the infrared sensor, though not able to measure distance accurately, has good angular resolution in detecting the absence or presence of an object. By using both sensors to scan a room, the robot is able to build a better map.

The infrared sensor is able to find edges of doorways and narrow passages that would be otherwise blurred by the sonars. The boundary of data points which is initially created by the sonar readings is redrawn appropriately to mark the doors detected by the infrared. The curvature primal sketch [Brady 84] is then extracted from this modified boundary, and significant curvature points are used as landmarks for matching between scans as the robot moves. Scans from subsequent moves can be merged and a room map boundary is created. This is then transformed into a list of polygons in order to provide the necessary input for path planners based on generalized cones [Brooks 83] or configuration space [Brooks 85].

The robot used in this work was Robart II [Everett 85a], Figure 1-1, built by LCDR Bart Everett, Director Office of Robotics and Autonomous Systems for the Naval Sea Systems Command, Washington, D.C. Robart was designed as a sentry robot and was loaned to the Naval Surface Weapons Center, Silver Spring, MD as a mobile platform for research and evaluation of sensors and navigation algorithms.

The sonar sensor was a Polaroid ultrasonic transducer [Polaroid 84] and the infrared detector was designed and built by LCDR Everett. The infrared sensor had four levels of power output and four stages of detector sensitivity. A description of the sensors and an analysis of their limitations is given in Chapter Two. Chapter Three displays examples of these limitations in plots of actual experimental data, formulates rules for combining the data, and shows how a better map can be extracted than if either sensor were used alone. In Chapter Four, this modified map is converted into another representation, the curvature primal sketch. Significant changes in curvature are used as landmarks for tracking between robot moves. A model of the room is then built up from a succession of these scans. Chapter Five illustrates how this representation is used as input to a planner and examples of two planners which use the senory data as input are given.

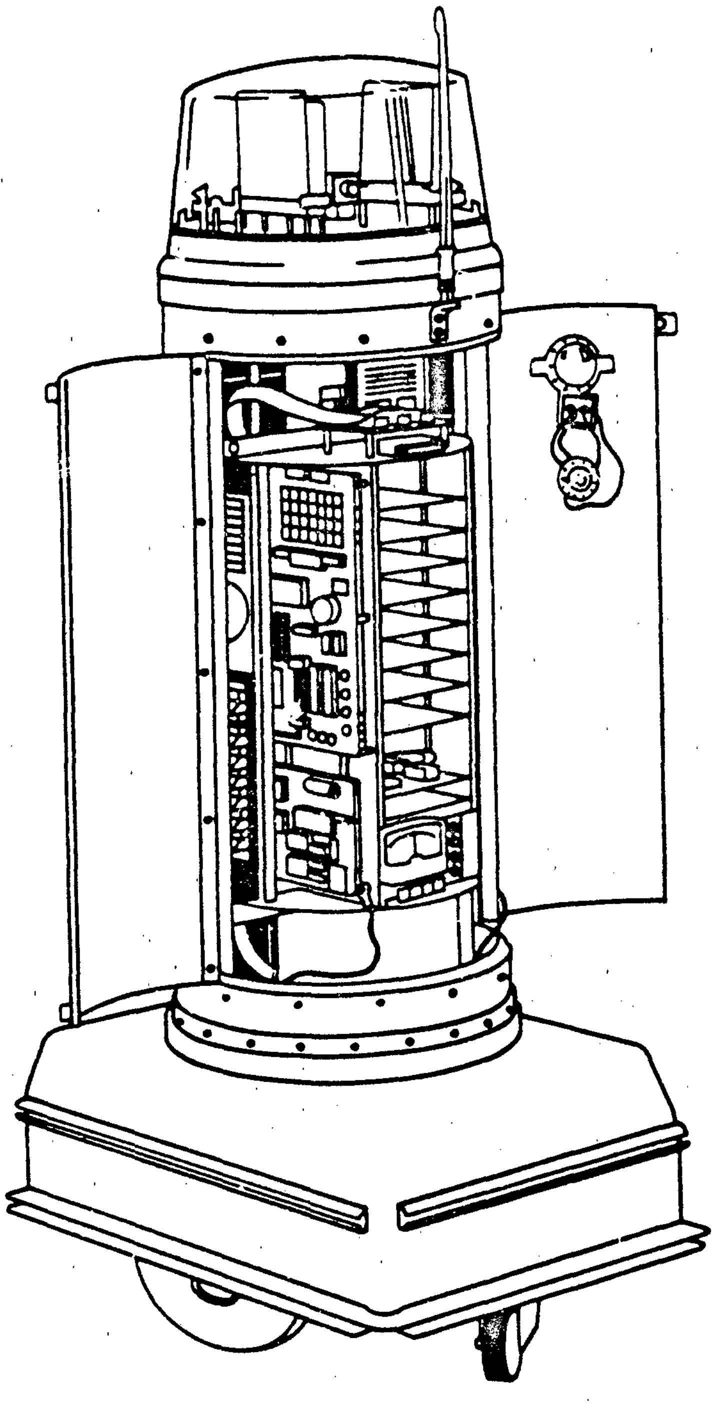


Figure 1-1:Robart II - An Autonomous Sentry Robot

Chapter Two

Related Work on Intelligent Mobile Robots

Starting in the late sixties, a few large projects were begun in the United States, France, England and Japan to develop autonomous robots. However, funding was gradually reduced as these projects failed to produce all that they had proposed. Researchers then began to focus on many of the subproblems that the early work had exposed to be much harder than expected: problems such as vision, natural language and planning in uncertain environments. Since then, there has not only been tremendous progress in image understanding, natural language and planning, but major breakthroughs in microelectronics to the extent that we now have massively parallel computers with which to tackle these problems in real time. Consequently, research is again picking up on the task of integrating all these modules to produce truly intelligent autonomous vehicles.

The survey which follows outlines the work which has been done in the past on mobile robots and summarizes some of the projects being pursued now. Special emphasis is placed on how these endeavors have tackled or solved the problem of building a memory map and using such a model for the purposes of navigation.

2.1 Shakey 1967-1969

Some of the earliest and yet at the same time most sophisticated work in applying artificial intelligence to robots was done at the Stanford Research Institute in the late sixties on an automaton named Shakey [Nilsson 69b, Coles 69]. Shakey, Figure 2-1, operated off a large time-sharing computer, an SDS 940, by radio link and had both a FORTRAN executive for control and I/O, and a LISP executive for maintaining its world model.

Its main sensor was a rotatable camera, and with this sense of vision and its many levels of software, it was able to navigate, explore and learn. This was some of the earliest

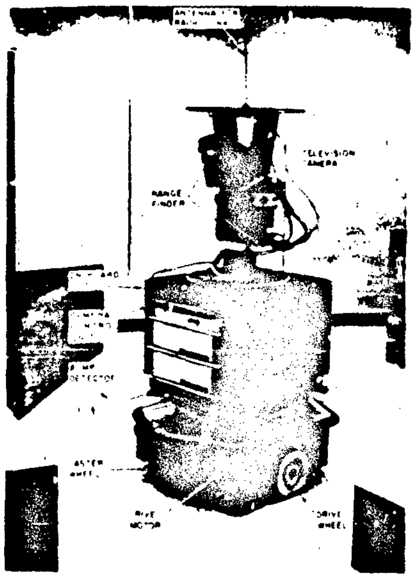


Figure 2-1:Shakey - Moravec 81b

work in machine vision, and one lesson learned was that vision was a very hard problem. Shakey also had natural language capability. A person could type in an English sentence command, and Shakey would parse the sentence and call up the appropriate FORTRAN or LISP programs to carry out the command.

Shakey's view of the world came from two models: a grid model and a property list model. The grid model divided the room up into nested 4x4 arrays called cells, where each element of the array was called a square. The entire world consisted of one cell, in which each square could be marked as full, partly full or empty. Partly full squares could then be represented as cells and further subdivided into 4x4 arrays of squares. Thus the room could be resolved to any desired level of detail, while its representation would require only a minimal amount of computer memory. This idea later evolved into the quadtree representation which is still often held to be a desired representation for work in recognition and planning. From the model, obstacle-avoiding trajectories could be calculated as shown in Figure 2-2. It was more difficult however, to plan journeys by using the grid model than by using a fully divided large array [Rosen 68]. Additional information had to be

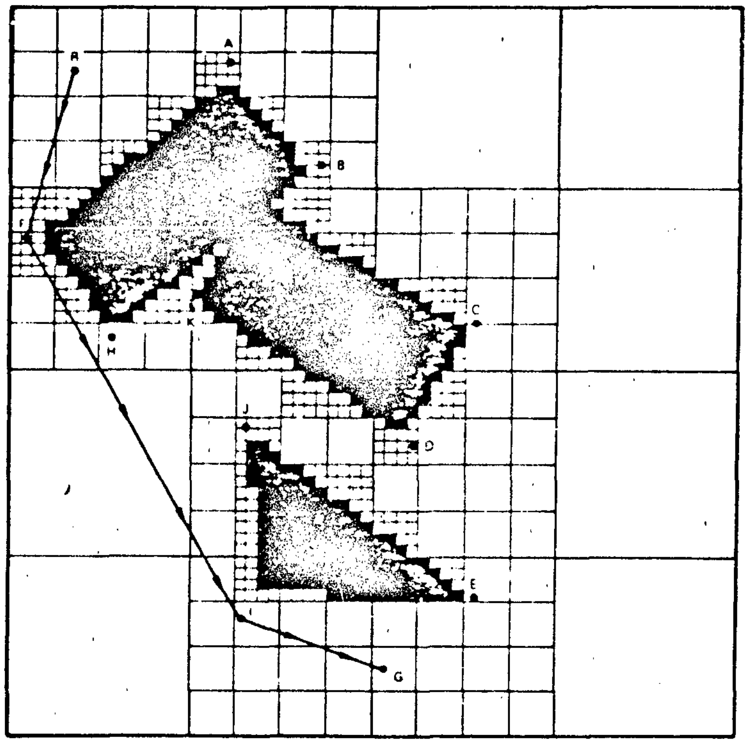


Figure 2-2:Shakey's Grid Model - Nilsson 69a

maintained to help programs using the grid model, such as depth of the cell in the model, coordinates of the cell, lengths of the sides, and pointers to parent squares or cells.

Vision was used as an input to the grid model. The camera would take a picture, convert it to a line drawing, determine floor boundaries of objects, and calculate free floor space. It would then add full and empty areas into the grid model.

One problem was that the robot's position was "dead reckoned" by keeping track of wheel rotations, and errors due to slippage caused Shakey to miscalculate its position. This forced the vision system to incorporate objects incorrectly into the grid model. Because of this, it was noted that effective reorientation techniques would be an important area for future study.

Although the grid model was usable for journey planning when the robot was only concerned about free or empty areas, the grid model was not suitable for other functions such as object identification.

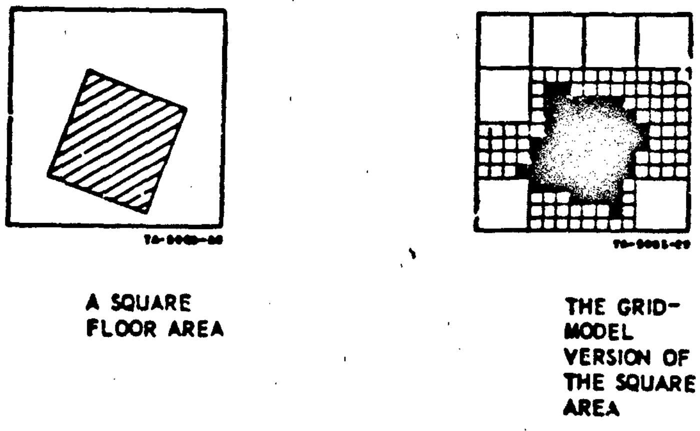


Figure 2-3: The Grid Model Cannot Clearly Represent the Obstacle as a Square - Rosen 68

As seen in Figure 2-3, the jagged edges in the grid model's representation of the square made it hard for the robot to recognize it as such. To solve this problem, a line model was proposed in which visual images would be processed into line drawings and a straight-line representation of obstacles would be used for a model. This was not successful, however, due to the inability of vision systems at that time to provide the accuracy needed.

In addition to the grid and line models, a property list model was utilized. The property list model, later becoming the n-tuple model, represented objects in terms of their properties, using LISP type data constructs. Thus an object somewhere in the room might be denoted as an ordered list of such features as x-coordinate, y-coordinate, angle, size, shape, etc. The property list model was used for interpreting commands such as "GO TO A BOX". The coordinates would be looked up under an object named "BOX", then the grid model would be accessed by FORTRAN routines to determine collision-free paths and to carry out the task.

The integration of the hierarchical levels of software gave Shakey the sophistication to remain the state of the art robot for many years. What is odd, is that Shakey, at the time, was considered a failure or at least an example of something the AI community had promised but couldn't deliver - namely, a completely autonomous robot. Shakey's environment had to be very simple for all his systems to work, and he was very slow, and well, "shakey". Funding on mobile robot research diminished and sponsors became disenchanted with AI in general for various reasons [Dreyfus 79]. This was mainly due to a change of plans at the Defense Advanced Research Projects Agency and not for scientific reasons.

The main lesson learned was that the instinctive skills which are easy for humans, such as seeing, moving, etc., are very hard to program into a robot, whereas higher level functions that are hard for humans, such as calculating, are much easier for a robot.

One of the contributors to the Shakey project was once asked if all the work that went into Shakey could have been done in software as a simulation. His answer was negative, because they wouldn't have known what to simulate. The difficulty lay in designing algorithms for poor data, not for perfect data, and they would not have known in which ways the data would have been poor [Raphael 68].

After Shakey, funding was continued in the areas of vision, natural language processing and planning as serious problems in and of themselves, and not necessarily as subproblems of a mobile robot system.

2.2 The Jet Propulsion Laboratory Mars Rover 1970-1973

In the early seventies NASA began a project to develop a rover to be used in planetary exploration [Lewis 73, Dobrotin 77, Miller 77, Lewis 77, Thompson 79]. It had been noted in previous Viking missions that due to long telecommunication delays it had taken several days to move a rock. Advantages sought in an autonomous robot would be reduced cost in both time and money for future space missions.

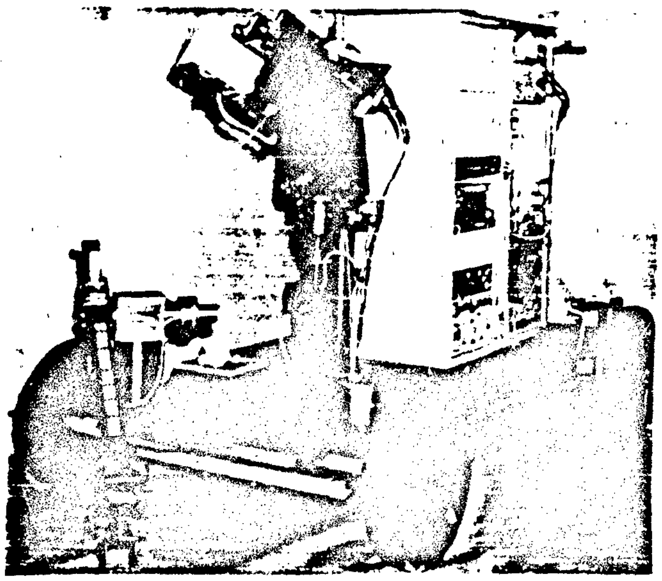


Figure 2-4:The JPL Mars Rover - Moravec 81b

The JPL robot, Figure 2-4, consisted of a mobile vehicle equipped with a six-degree-of-freedom manipulator (a modified Stanford arm) and an assortment of sensors (laser range-finder, stereo TV cameras, tactile sensors and proximity sensors). The navigation system used a gyrocompass and optical encoders on the wheels for dead reckoning. An on-board mini-computer (General Automation SPC-16 with 32K memory) for real-time control of motors communicated with a remote PDP-10 on the Arpanet. The remote system was used to process TV and laser pictures, to construct the "world model" and to do planning and decision making. The robot, however, never advanced beyond the stage of being tethered with a 50-100 foot cable.

The rover's objectives were to analyze a scene for traversability, plan a path to the goal and follow that path without bumping into anything. These objectives were achieved only in a simplified environment consisting of a laboratory with a flat surface, a limited number of obstacles and constant illumination.

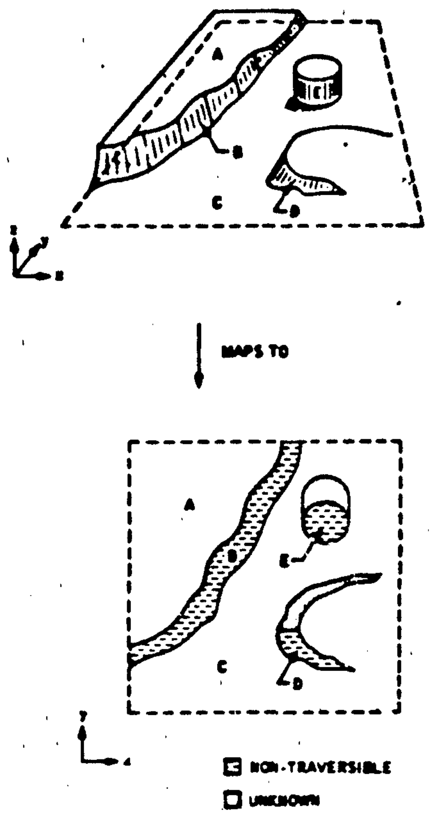


Figure 2-5: The JPL Rover's Map - Thompson 79

The model of the world held by the JPL Rover was a segmented terrain model derived by inputs from the vision system. Since the area explored by the robot was large, the terrain model was partitioned into map sectors of a convenient size and stored as separate files. Each sector was a fixed lattice of grid lines drawn parallel to the Rover's absolute coordinate system. The resultant collection of map sectors was similar to a catalog of charts. Each map sector represented areas that were either not traversable or unknown, as shown in Figure 2-5. All other areas were assumed traversable. Non-traversable regions were described as boundaries of polygons and these regions were then represented as lists of the vertices of those polygons.

This map had to be continually updated while the robot moved around performing its assigned task, and errors frequently got incorporated into the model. The first source of error was the uncertainty in vehicle position due to dead reckoning. This error increased with the distance from a known location. The second source of error was the limitation of the vision system to accurately determine relative positions of obstacles. Once an internal model was built, the Rover could refer to that model and using various search algorithms, plan an optimum route to its goal.

Although the JPL Rover project was able to produce several useful robotic subsystems such as the manipulator, the laser rangefinder and the navigation system, putting them together did not result in a completely autonomous robot as desired. The tether still remained and improvements were still needed to reduce errors in the respective subsystems so that the final system would be able to act intelligently and with a higher level of coordination. It was the classic case of an attempt at system building before the technology for the components was available.

2.3 The Stanford Cart 1973-1981 and CMU Robots 1981 -

From 1973 to 1981, work was done at the Stanford University Artificial Intelligence Lab by Hans Moravec on developing a remotely controlled TV equipped mobile robot [Moravec 81b, Moravec 83]. A crude cart was used as the mobile platform, but a sophisticated vision system and appropriate navigation and obstacle avoidance software enabled the Cart to move through cluttered spaces.

The Cart with its camera system is shown in Figure 2-6. The Cart used stereo imaging to locate objects and to deduce its own motion. A TV link connected the Cart to a remote KL-10, which sent control commands to the Cart and also did all image processing. The camera on top of the Cart was mounted on rails and slid by remote control to nine different

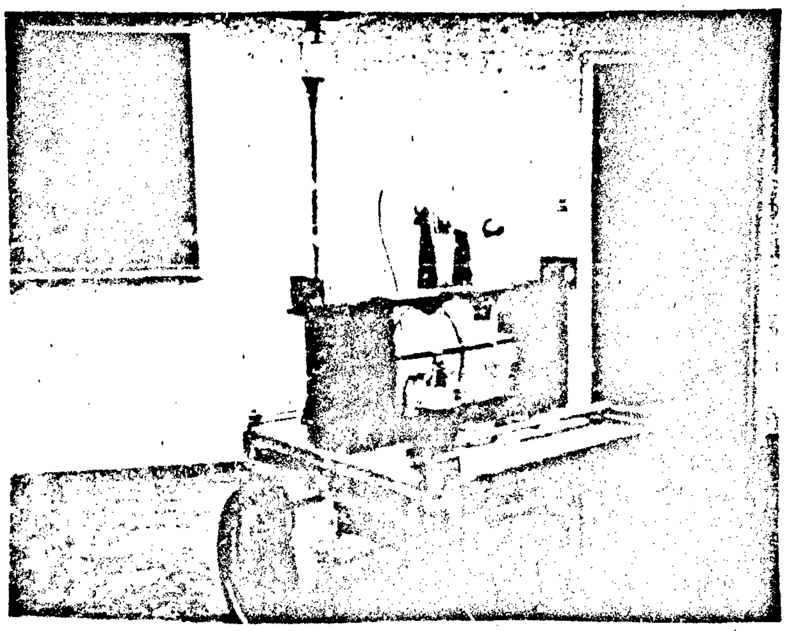


Figure 2-6: The Stanford Cart - Moravec 81b

positions to get nine pictures of the view before it. These pictures were then digitized and processed to extract 3D information from the scene.

Processing of the pictures amounted to extracting features from each picture and then correlating those feature points between any two images. Features were extracted by running an "interest operator" over each digitized picture, which would pick out areas in the picture which had the maximum gradient of grey scale. Thus points such as the corner of a table would be picked out because the top of the table might be well lit while the side was dimmed by shadow. Feature points would be marked in as many of the nine pictures as possible and then a correlator routine would compare that feature point's change in pixel position between any two pictures. Knowing that information and the distance that the camera had moved gave distance to the object. Nine pictures were used to increase reliability.

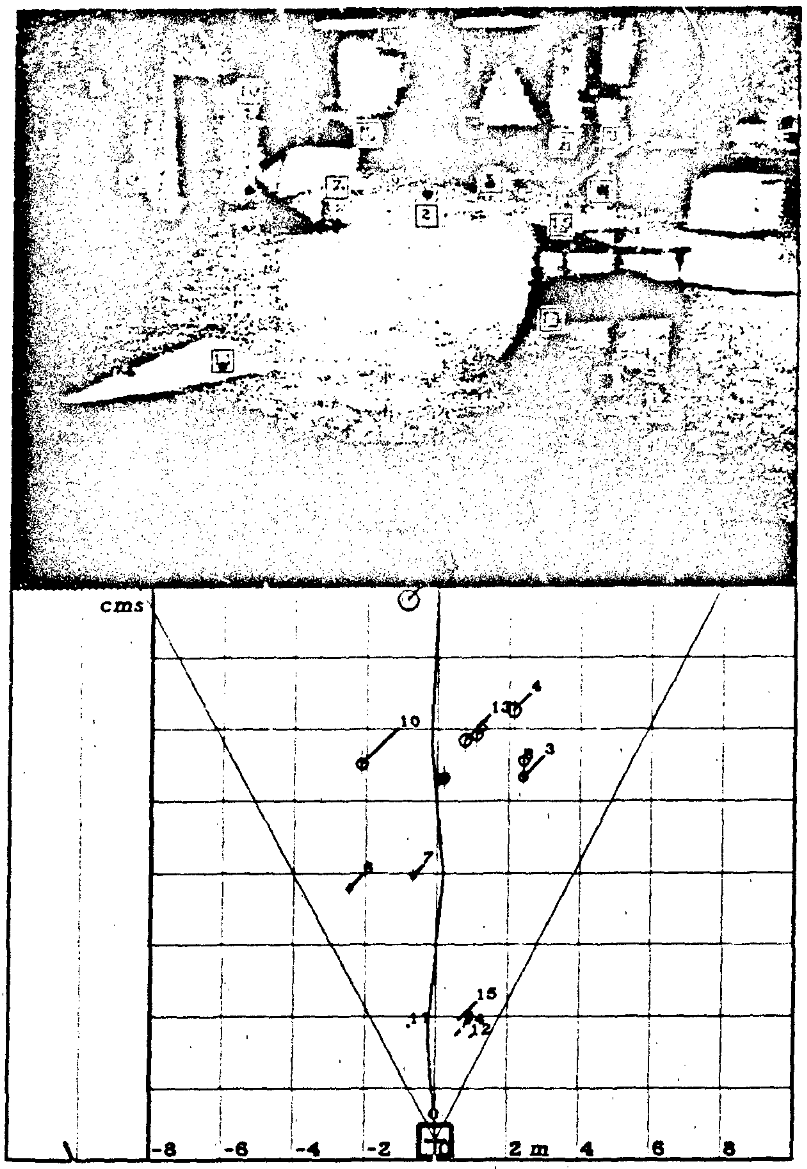


Figure 2-7: The Cart's View of the World - Moravec 81b

The digitized image with its feature points marked is shown in Figure 2-7. Also shown is the path which the Cart had planned to reach its goal. This information was used to build a model, and from this model it would plan an obstacle avoiding path to its destination. The system worked but was slow due to many factors. These ranged from the many computations needed to deduce the cart's own motion since its own dead reckoning system was so weak, to the fact that the system utilized interpreted LISP running on pre-LSI technology. The Cart would move one meter, stop, take pictures, think for fifteen minutes, and then move forward another meter. The Cart successfully maneuvered through several 20 meter courses (each taking about five hours) but failed in other runs.

Some problems in these runs were that featureless objects were hard to see, and also that shadows often moved considerably during the course of the run, throwing off the correlator since shadows produced new feature points due to their high contrast. Another problem involved weaknesses with the vision system's ability to maintain an accurate self-position model. Although the model was to be updated after each lurch, small errors in the measured feature positions sometimes caused the solver to converge to a position with an error beyond the expected uncertainty. Any features incorporated into the model after the Cart lost its correct sense of self-position were inserted wrongly. These errors were cumulative and caused the same object to seem to be in another place. The combination of old and new positions of these objects made it appear to the Cart that the path was blocked when in actuality it was open.

Much of this research has continued at Carnegie Mellon University in a number of systems which they have built there. Figure 2-8 shows one type of world model they use to represent sensory data [Moravec 85]. Twenty-four Polaroid sonar transducers are mounted in a ring around the robot and the data from separate sensors are combined in a probability map which represents areas that are either empty, occupied or unknown. Each cell in the grid represents six square inches of floor space, and the value in each cell can range from -1 to 1. Negative numbers represent a probability that the area is empty, while positive numbers mean it's probably occupied. Zero represents the unknown. The basic idea is that because of the wide beamwidth of the sonar, there may or may not be an object in the line of sight: some object might be picked up by the edges of the sonar beam. In addition to a probabilistic measure in terms of angle, range can also be modeled. For any sonar range reading there might be some small error, but most of the area up to that distance can confidently be marked as empty. By combining information from many readings as the robot moves through the room, areas known to be empty or occupied are expanded, and the uncertainties associated with each region are decreased. The effect is that object locations become known with increasing precision.

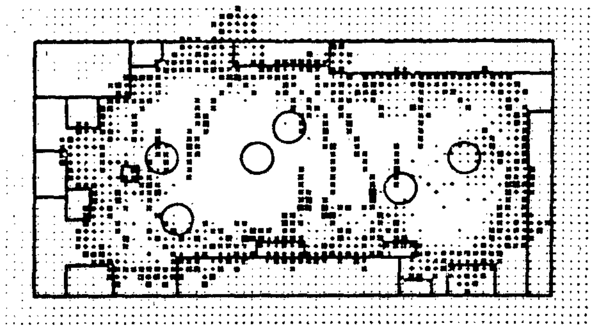


Figure 2-8: Empty, Unknown and Occupied Areas in a Sonar Map - Moravec 85

2.4 Hilare 1977-

Work began in 1977 in France at the Laboratoire d'Automatique et d'Analyse des Systemes to develop an autonomous robot that utilized multiple sensors and would be equipped with a multi-level computer and decision system [Briot 81, Bauzil 81, Ferrer 81].

Hilare, Figure 2-9, has a 3D vision system which uses a laser range-finder in conjunction with a video camera. Its sensor system also incorporates ultrasonic devices as proximity detectors for close-in obstacle detection and for paralleling a wall. It uses a system of infrared beacons mounted on the walls in the corners of its room to give it absolute positioning information. This works by using two infrared emitters and detectors on the robot. Measurements of angles are made by counting control pulses. The multi-level computer system consists of three 8085 on-board microprocessors for sensory data

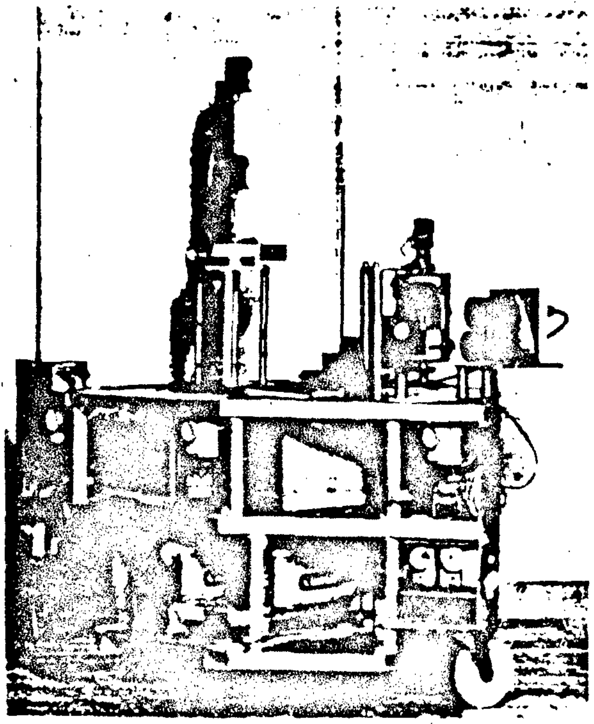


Figure 2-9:Hilare - Ferrer 81

processing, an off-board MITRA-15 minicomputer for navigation and communication tasks, and a remote IBM-370 used as a peripheral to the minicomputer for complex tasks.

A distributed decision-making capability is provided through a system of cooperating expert modules which have expertise in the areas of object identification, navigation, exploration and planning. These modules consist of specialized knowledge bases, algorithms and heuristics, error processing capabilities, and communication procedures. This system enables Hilare to carry out navigation tasks which involve universe modeling, building a plan, and supervising the development and execution of that plan [Giralt 77, Laumond 83]. Hilare's world model defines obstacles as polyhedrons whose projections on the floor determine the navigation problem. This model can either be determined by the robot's perception system or provided as initial information.

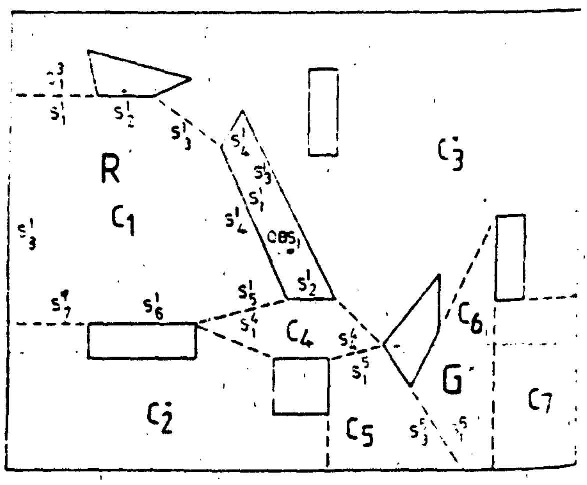


Figure 2-10: Hilare's World Model - Giralt 77

The obstacles are represented as an ordered list of segments where each segment is represented by the Cartesian coordinates of its leftmost point, an angle with respect to some reference axis, and its length. As seen in Figure 2-10, empty areas are partitioned and represented as convex polygonal cells which include obstacle segments. Trajectories within cells are straight line paths between entry and exit segments so that adjacent cells have common segments which are traversable by the robot. This pattern of connectivity can then be represented as a graph, which provides the structure necessary for path finding.

Optimum paths are determined by making a search over the resulting graph while minimizing costs in terms of distance and energy requirements. The minimization function is a linearly weighted combination of path length, angle of planned direction change, and the number of predicted stops, together with a term which accounts for the uncertainty of information obtained by the robot and also the path viability due to estimated obstacle clusterings.

The model is built up by merging information from laser rangefinder scans as the robot moves from one position to another [Chatila 85]. Perceived obstacles are assigned their own coordinate frames. Before the robot moves, it predicts what it might see. That is, it hypothesizes which edges might become occluded and which might become uncovered. Then after moving to its new location, it matches the predicted model to what it actually perceives. If some adjustment must be made to bring the two into alignment, that that can be used to update the robot's position. Figure 2-11 shows four figures (from left to right respectively) that depict the process of perceiving, predicting, perceiving and merging the models.

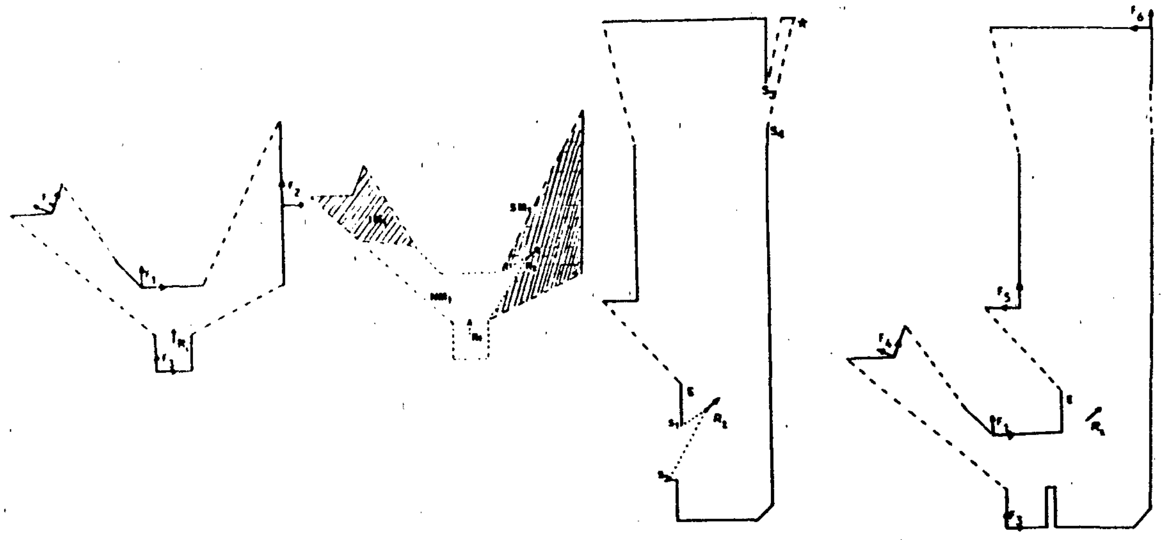


Figure 2-11:Merging Perceived and Predicted Models - Chatila 85

2.5 Robart I 1980-1982

Robart I was built at the Naval Postgraduate School by LCDR Bart Everett, to serve as a feasibility demonstration for an autonomous robot [Everett 82b, Everett 82a]. Robart, Figure 2-12, would randomly patrol a house sensing for fire, smoke, flooding, toxic gas, intrusion, etc., and take appropriate warning action if any of these conditions were found. The goal of the project was to show that certain applications could indeed be handled by autonomous mobile robots, using current technology, under the right conditions. The particular application of a sentry was chosen because it did not require any end effectors or a vision system. The project was done on an extremely limited budget, using simplified

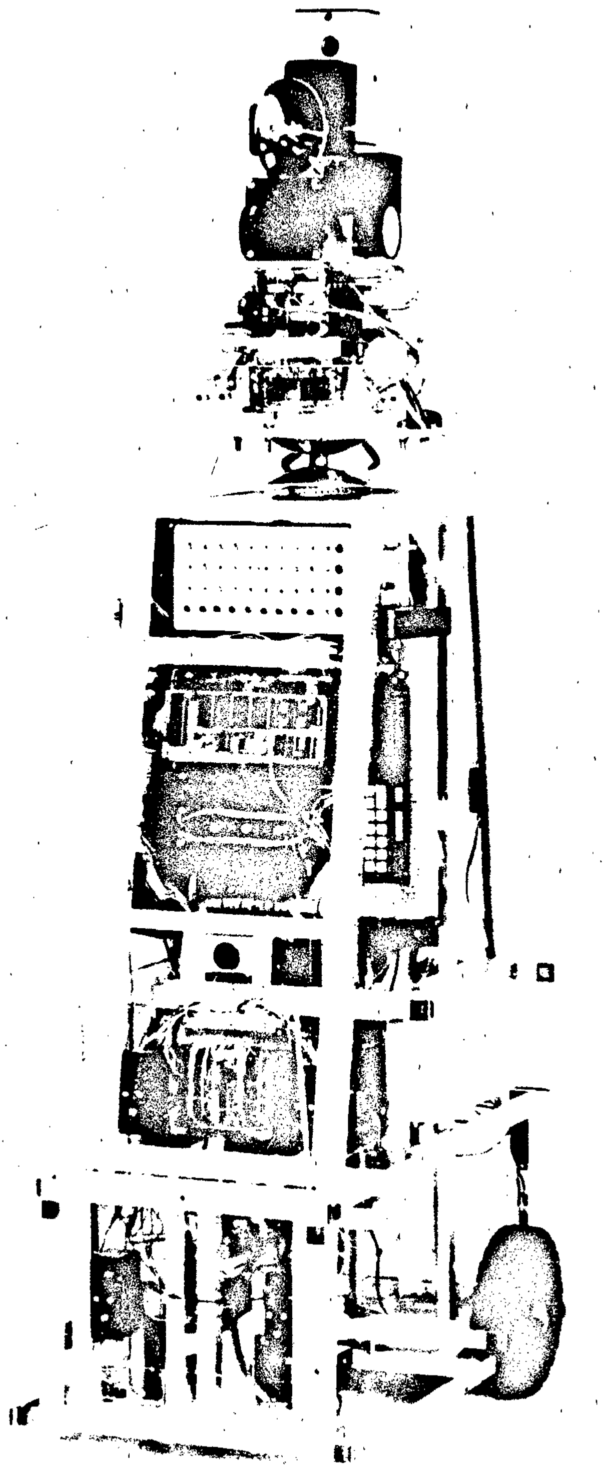


Figure 2-12:Robart I

approaches: the philosophy being that if successful under those conditions, an extrapolation should show the tremendous potential if later addressed with sufficient funding.

The robot had a single forward looking ultrasonic ranging unit, a long range near-infrared proximity detector that could be positioned by a rotating head, ten short range near-infrared proximity detectors, and tactile feelers and bumper switches for collision avoidance. The battery voltage was constantly monitored and when it fell below a certain adjustable threshold, the robot would activate, via a radio link, a homing beacon placed on top of its recharging station. For simplicity, an ordinary 75 watt light bulb was used as the beacon, tracked by an optical photocell array located on the robot's head. Thus the head position represented the relative bearing to the beacon, and the robot could home in on the battery recharger. The software provided verification of the correct beacon acquisition, the ability to maneuver around obstructions enroute, and the correction of any misalignment that occurred as a result of collision avoidance.

Other sensors onboard included a true-infrared body heat sensor which could detect a person out to a distance of fifty feet. This sensor was fairly directional, and mounted on the head so as to be positionable under software control. Also mounted on the head was a near-infrared long range proximity sensor with a parabolic reflecting collector, able to detect the edges of an open doorway to within an inch at a distance of six feet. This angular resolution allowed the robot to steer toward the center of the doorway while still some distance away.

In addition to its multitude of sensors, the five-foot-tall Robart could also speak. Voice synthesis was not only used to warn of the presence of intruders or other alarm conditions, but could also report on the internal status of its circuits, system configuration errors, time-of-day, temperature, etc.

Robart's behavior appeared arbitrary, or at least not preprogrammed. An operating system provided for the selection of various behavior primitives, each designed to meet a specific goal, based on the output of specific sensors, via interrupt software. When no specific actions were called for, a routine was randomly chosen from a preprogrammed set of sixteen routines that filled in the gaps. Some of these routines would move the robot more or less randomly to a new vantage point, where it might elect to stop and re-enter the surveillance mode. Motion under these circumstances usually involved moving straight ahead, unless it saw an object, in which case it would swerve to one side or the other as appropriate. It would then continue moving in the new direction until it encountered another obstacle.

Robart could also be put in either the "Hostile" or "Friendly" mode. In the "Friendly" mode it would greet a person with an amiable "Hi" or "Hello", while in the "Hostile" mode it would announce "Intruder, Intruder", and then advise the intruder to leave the room.

All sensors were interfaced to one 6502 based SYM-1 computer on an interrupt basis. A triangular wheelbase was utilized, with the one front wheel providing power and steering. Optical encoders were not used so dead reckoning was not performed, but an A/D converter gave four bits of information on steering wheel angle. The rotating head had similar resolution. In the worst case, wheel and head together could have as much as 22 degrees of error when looking for the recharging station. This was done on purpose, however, to demonstrate the feasibility of software compensation. In over 200 dockings, Robart only failed once to hit its recharging station within half an inch from the centerline of its front bumper. The entire robot was powered by one 12V 20 amp-hour battery, providing roughly ten hours of service, with fourteen hours needed for full recharge.

2.6 Robart II 1982 -

Robart II [Everett 85a], the robot used in this thesis, is an improved version of its predecessor, Robart I. Robart II is a battery operated autonomous mobile robot which stands four feet tall, and measures 17 inches across at the base. The number of on-board 65C02 (CMOS) based computers has been increased from one to eight, allowing for more sensors to be interfaced and for more processes to run in parallel.

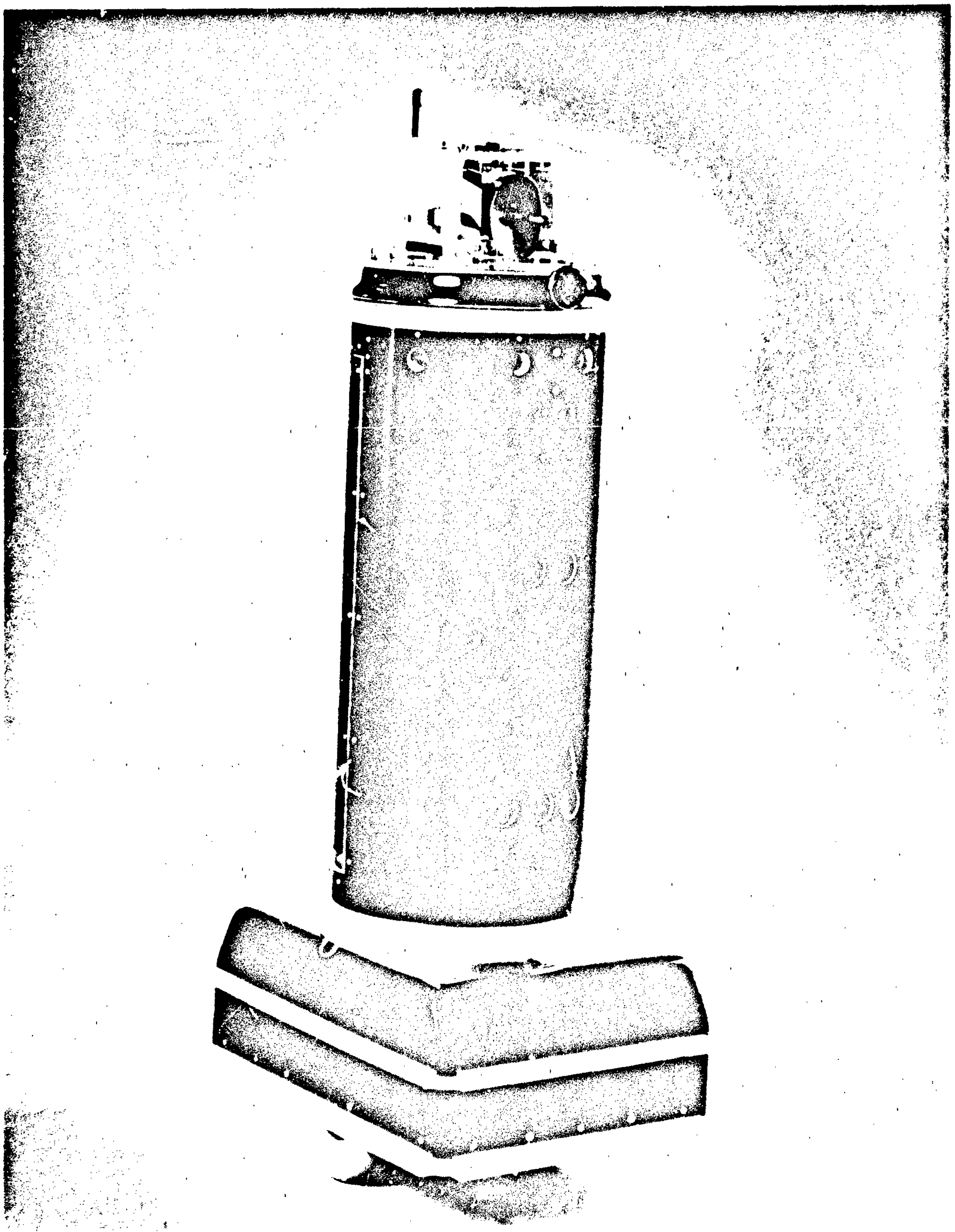
The platform houses a variety of sensors for path planning, collision avoidance, and environmental awareness. These include six ultrasonic rangefinders, fifty near-infrared proximity detectors, a long range near-infrared rangefinder, plus various sensors used to detect special alarm conditions, such as fire, smoke, toxic gas, flooding, vibration and intrusion. Four true infrared motion detectors are employed for detecting the presence of an intruder up to seventy-five feet away, reacting to the thermal radiation emitted by the human body. Special internal circuitry checkpoints are analyzed by self-diagnostic software and, when necessary, operator assistance is requested through speech synthesis.

A front view of Robart II (Figure 2-13) shows the five sonar transducers on the body and one on the rotatable head. The long-range near-infrared sensor with parabolic reflecting dish sits on top of the head and three of the true infrared motion detectors are mounted just below the head. A rear view (Figure 2-14) exposes the card cage which houses the eight computers and all the interface circuits.

Two twelve-volt DC motors powering eight-inch diameter wheels are mounted on either side of the base platform so as to be symmetrical with respect to the vertical axis of the robot, and are independently controlled to provide differential steering. Two non-driveable casters are included for stability.

The computer architecture onboard consists of a SYM-1 6502 board acting as a Scheduler for the five dedicated MMC-02 65C02 controllers. These dedicated controllers directly interface with the head, the drive motors, the sonars and the speech synthesis and recognition processors. One controller also controls two linear CCD-array cameras which are still to be installed. The SYM-1 manages all communications between processors and holds global information which several controllers may need. All low level controllers receive commands from the SYM-1 via an eight line parallel bus, and communicate information back up via a common serial interface.

A second low level dedicated 65C02 controller is used to operate six ultrasonic ranging modules through a special multiplexing circuit [Everett 85b]. Five of these units have their transducers arranged in a forward-looking array, with overlapping beam patterns. These transducers can be sequentially fired in any combination, as determined by the command



[illegible] Part II - A Front View

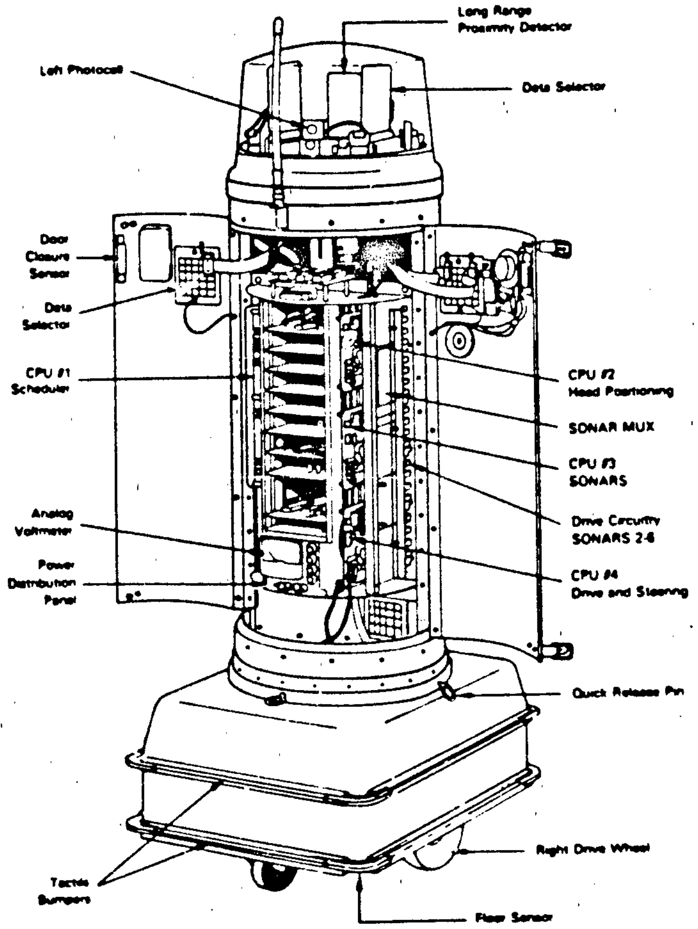


Figure 2-14:Rear View - Controlling Computers - Everett 85a

from the SYM-1. Associated ranges are then fed back up to the Scheduler. A sixth transducer is mounted on the rotatable head, positionable up to 100 degrees either side of centerline. The position and velocity of the head are controlled by another dedicated low level microprocessor. The head controller also interfaces to the A/D converter which monitors the environmental sensors and the proximity detectors.

A fourth dedicated controller is assigned the function of controlling a DT-1050 microprocessor based speech synthesizer, and a speech recognition board which can

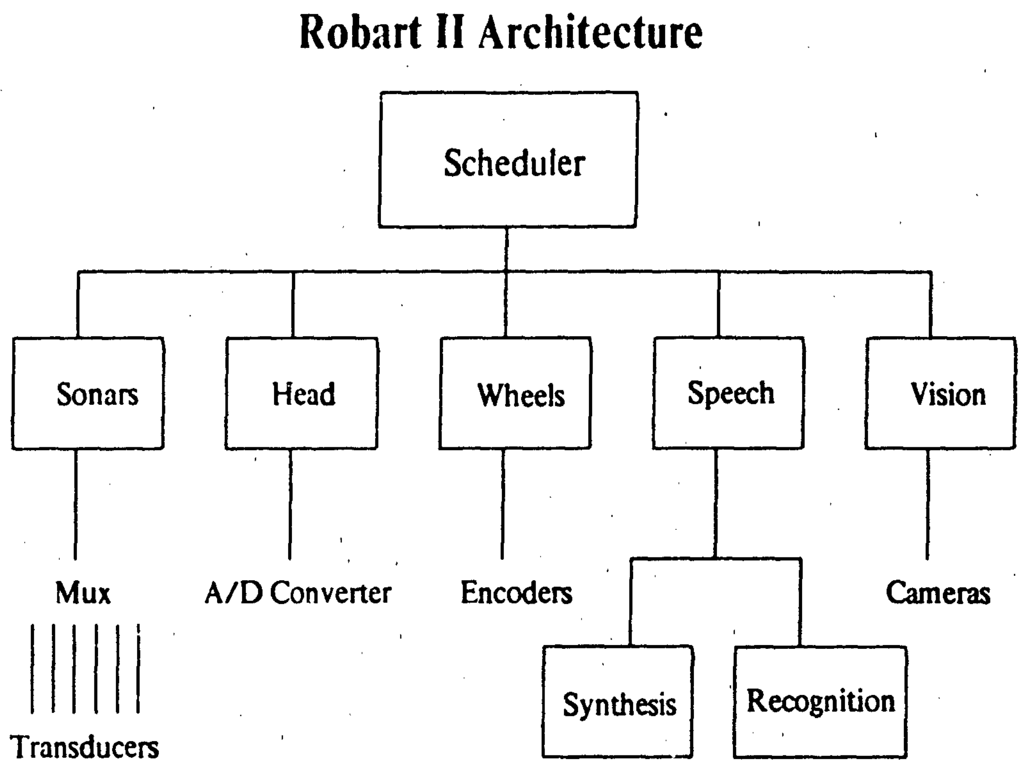


Figure 2-15: Architecture of the Controllers

recognize up to 256 speaker-trained words. The fifth low level processor controls the drive wheels through pulse width modulation. The two wheels are synchronized by encoders on the drive shafts. The encoders also supply distance and velocity information for dead reckoning.

An off-board IBM-XT was used to gather the sonar and infrared data presented in this thesis. The SYM-1 Scheduler directed the head and sonar controllers to fire the infrared and sonar sensors once each as the head moved through 256 positions. All readings were returned to the SYM-1 and shipped out to the XT, which was running a C program to capture the data. Forty-six sets of scans were taken as the robot was positioned throughout the room, and all this data was transported to a LISP Machine at MIT for further processing.

Much of the structural hardware on Robart II was already assembled when I began my cooperative education assignments at Naval Surface Weapons Center. One computer had been installed and a few sensors had been mounted and interfaced. Nevertheless, LCDR Everett and I spent uncountable hours and many weekends over the next year hacking circuits, software, vendors and packaging, to bring eight computers on line, make them all talk to each other, control the drive wheels and head, interface with the sensors, and make Robart talk and follow us around. In the end, we had a very beautiful and elegant robot.

Chapter Three

Modeling the Sensors

The survey of the previous chapter shows clearly that much work remains to be done in integrating sensory data into robot systems that can produce intelligent action. In order for a robot to analyze the incoming data effectively, it must have a good model of the strengths and weaknesses of its sensors.

The Sonar Rangefinder

A careful look at the specifications for the Polaroid transducer [Polaroid 84] along with a few simple measurements initiate the following as a reasonable model. Figure 3-1 shows the radiation pattern for the transducer.

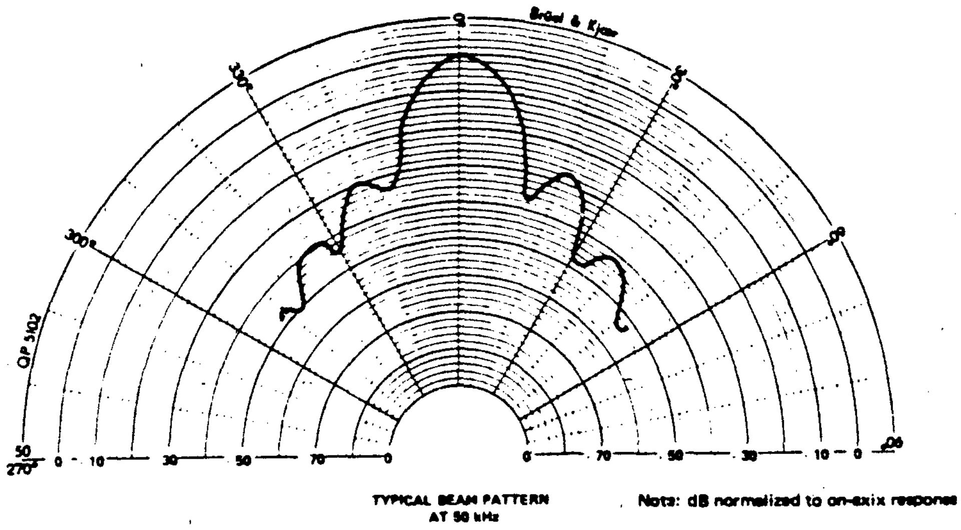


Figure 3-1:Radiation Pattern of the Sonar Transducer - Polaroid 84

The beamwidth, typically measured at the 3dB point, is shown to be roughly 10 degrees. However, in actual practice, the transducer is sensitive enough to detect echoes of energy transmitted from the sidelobes. In testing, the rangefinder could detect a one inch diameter pole up to an angle of 40 degrees. With such a large beamwidth, a robot scanning a room with this sensor perceives all objects to be much wider than they really are.

The rangefinder is capable of measuring distances to an object with a resolution of 0.12 inch through a range of 0.9 to 35.0 feet. This is accomplished by measuring the time of flight between a transmitted pulse and a returned echo and multiplying by the speed of sound. The distance measure, however, is not necessarily the distance in the direction the sensor is pointing, since the width of the beam may cause an echo from one edge to be returned before the echo from the centerline. Figure 3-2 illustrates that although the sensor is pointing in the di-ction along AB, the measured distance returned is actually AC.

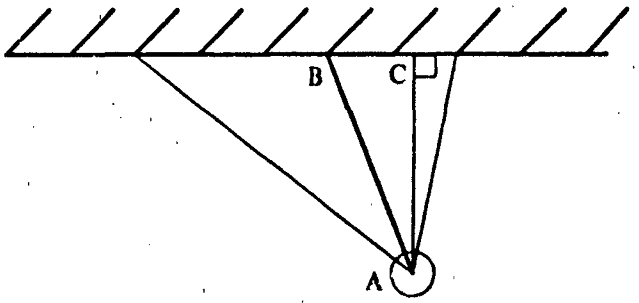


Figure 3-2:Sensor Measures Shortest Distance to a Wall

Another measurement error is due to specular reflections on smooth surfaces. Due to the large wavelength of sound, about 1/8 of an inch, many surfaces appear smooth. A sonar beam incident on such a surface does not reflect an echo directly to the sensor. Instead, it bounces off at an angle equal to the angle of incidence, and possibly bounces off other objects before being detected. Hence, the transducer measures a much longer distance than it should.

Figure 3-3 illustrates this problem. In actual tests against a smooth surface such as sheet rock, specular reflections occur when the sensor was aimed at an angle less than 25

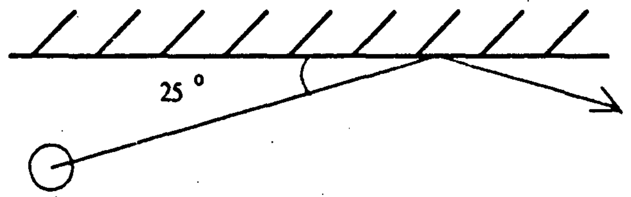


Figure 3-3: Specular Reflections Off a Smooth Surface

degrees from the surface. Against a rougher surface such as cinder block, there were no specular reflections at all.

Other errors come about due to atmospheric effects, such as the change in the speed of sound caused by temp- erature and humidity changes. The speed of sound is a function of temperature where:

$$\text{Speed of sound} = 331.4\sqrt{\frac{T}{273}}\ \frac{\text{meters}}{\text{sec}} \quad (\text{T in Kelvin})$$

Distances returned when assuming 80 degrees Farenheit, but where actual temperature is 60 degrees, will be seven inches too long [Everett 85b]. The effect of relative humidity can be found by table look-up [Maslin 83]. Robart carries onboard sensors for temperature and humidity, so these errors can be compensated for before the sonar data is sent on for processing with the infrared data.

Finally, there is another type of measurement error due to the form of the sound pulse which is transmitted. The pulse is actually a chirp, 1 ms long, of 56 pulses of 4 different frequencies. There are 8 pulses at 60 kHz, 8 pulses at 56 kHz, 16 pulses at 52.5 kHz and 24 pulses at 49.41 kHz. The time of flight measurement begins with the rising edge of the first pulse transmitted, and ends with the detection of the first echo. Figure 3-4 shows a timing

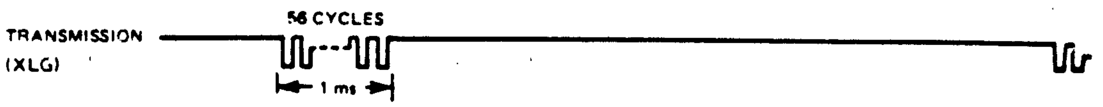


Figure 3-4:Transmitted Pulse - Polaroid 84

diagram. Four different frequencies are transmitted to compensate for the fact that different types of surfaces absorb energy of different frequencies. With four frequencies, it is more likely that every surface will return an echo. However, that echo is not necessarily associated with the first pulse, the one from which the time of flight is measured. In the worst case, an echo from the last pulse adds 1 ms onto the actual time of flight. This error corresponds to about 3 inches of additional distance measurement.

Modeling the Infrared Detector

The infrared sensor emits a pulse of infrared light and then uses a parabolic dish and an infrared detector to sense, over a very narrow area, any returned infrared energy. Unlike sonar, the time of flight of the light pulse cannot be easily measured. The sensor merely gives an indication of whether or not any returned pulse was detected. So, although distance to an object cannot be ascertained, the absence or presence of an object can be determined with very good angular resolution.

The infrared emitter used here is actually slightly more complicated than described above. Four LEDs are used to enable the sensor to incrementally step up the power output so that the range is increased. A good analogy is that of a blind man's telescoping cane. First one LED is fired. If no return is detected, two LEDs are then fired simultaneously, doubling the power output, and extending the range of the sensor. If, again, there is no detection, three LEDs are fired in unison, and finally four if necessary. When a returned pulse is detected, the robot notes how many LEDs were fired, and this gives a very rough indication of range. However, information which is more beneficial is to have the robot scan and note the point at which there is a discontinuity from detection to no detection, or vice versa. This often means a corner has been found, as shown in Figure 3-5.

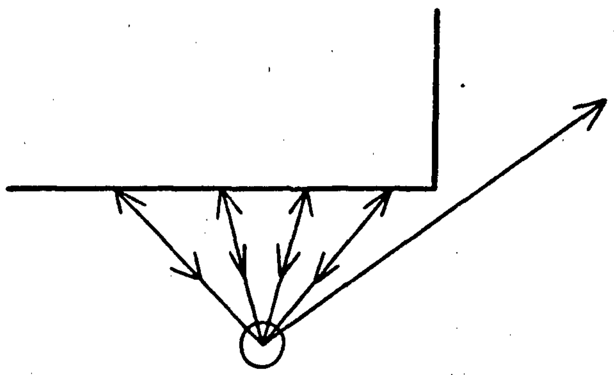


Figure 3-5:Infrared Sensor Finds a Corner

The infrared detector circuitry is also slightly more sophisticated than described above. Instead of a binary Detect/No Detect signal, there are actually four detectors of varying sensitivity. After the first single LED is fired, the robot polls these four detector outputs, starting with the least sensitive one. If any detector has sensed an echo, the robot notes how many LEDs were fired and which level of detector sensitivity was first to pick up the echo. If none detected an return, two LEDs are fired together, all the detectors are polled, and the process is repeated up to the firing of all four LEDs. Figure 3-6 shows a diagram of the infrared sensor.

The resulting data is input to the computer in a two-digit format, such as 31. The first digit signifies how many LEDs were fired and the second digit tells which detector circuit sensed the echo. In this example, three LEDs were fired and the first detector sensed the echo.

The maximum range of the sensor, using 4 LEDs, is approximately 18 feet, although only 3 LEDs were actually used here. They gave a maximum range of about 10 feet. The range is a function of how much energy is reflected off the surface. Different types of surfaces absorb varying amounts of infrared energy, so this sensor gives only a rough estimate of distance ranges. However, smooth surfaces do not pose the same problems of

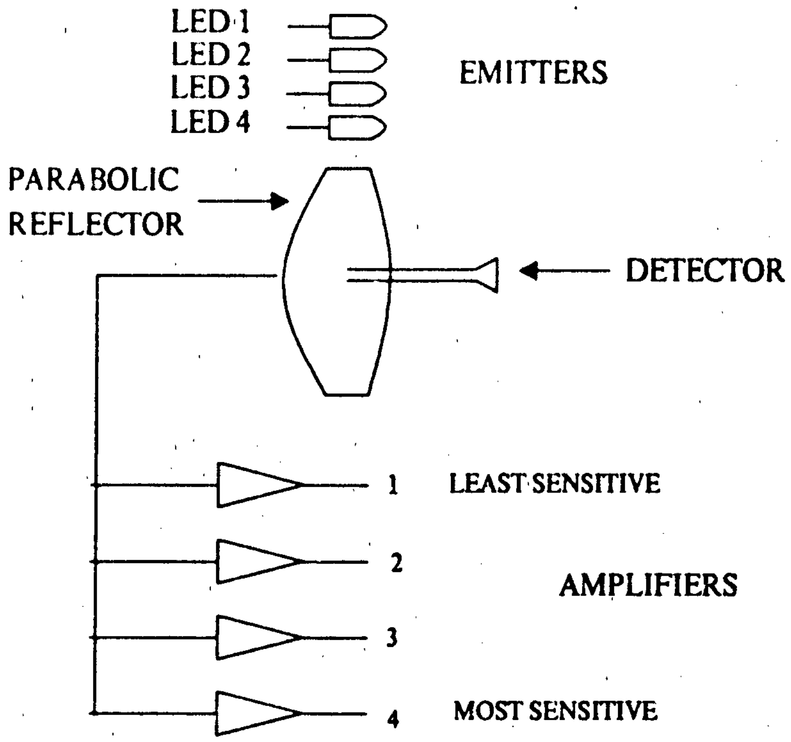


Figure 3-6:The Infrared Sensor

specular reflection as they do with sonars, because of the much shorter wavelength of infrared light.

Chapter Four

Combining the Data From Both Sensors

After defining a model of how the sensors work and taking experimental data, it's possible to formulate a set of rules for how the information from each sensor should be combined to build a model. These rules will tell when data from either sensor is valid and which sensor to rely on when they give conflicting information.

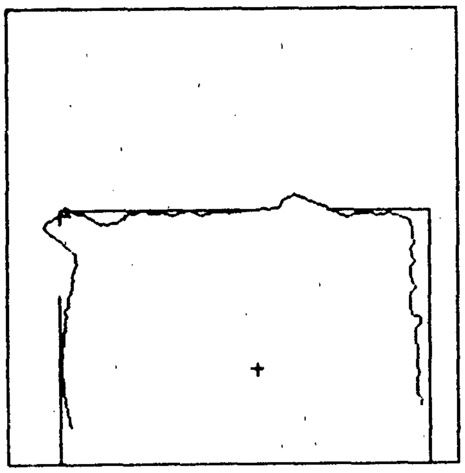
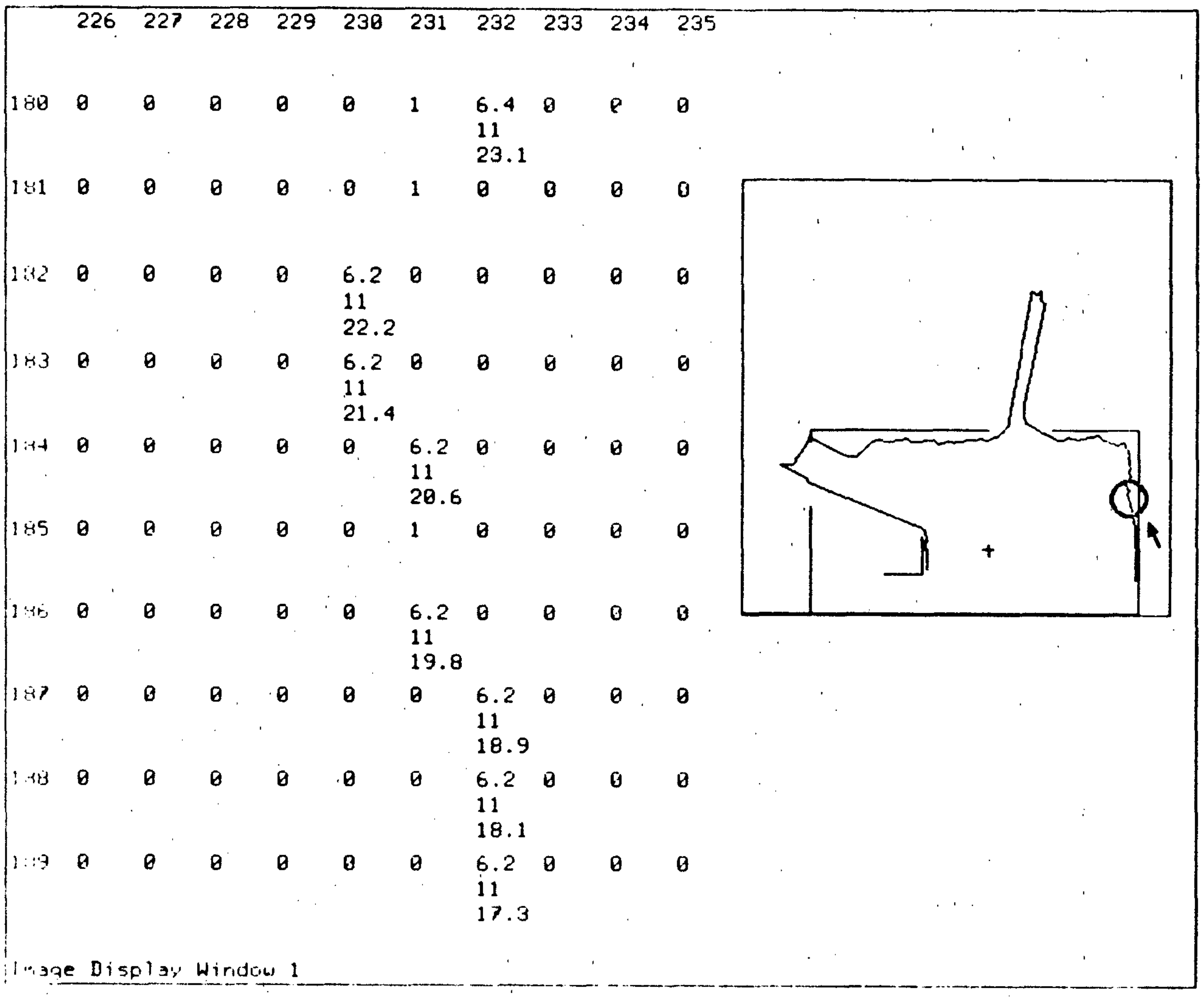


Figure 4-1: Sonar Plot of the Room

All the data was taken in a basement in which there were different configurations of obstacles and the robot was situated in various locations. Figure 4-1 shows one sonar plot of the room in which the robot was 6 feet in front of a wall that had an open door. Also, to its left, 7.5 feet away, was another open door. The room was 19' 7.5" long and 13' 8.5" wide and the walls were relatively rough, with cinder block along one side and exposed studs along the others. Consequently, the plot looks fairly clean. However, it's clearly seen that, even though the doors are open, they appear closed to the robot because at that distance, the sonar beam is wider than the door.

The sonar transducer and infrared sensor are both mounted on the robot's head, which rotates 100 degrees both left and right. Two hundred fifty six readings are taken on each scan while the robot is held stationary. The angle at which each reading is taken, and the distance measured are converted to cartesian coordinates and overlaid onto the actual room map. All the data is displayed in a 256 x 256 bitmap array.



	226	227	228	229	230	231	232	233	234	235
180	0	0	0	0	0	1	6.4 11 23.1	0	0	0
181	0	0	0	0	0	1	0	0	0	0
182	0	0	0	0	6.2 11 22.2	0	0	0	0	0
183	0	0	0	0	6.2 11 21.4	0	0	0	0	0
184	0	0	0	0	0	6.2 11 20.6	0	0	0	0
185	0	0	0	0	0	1	0	0	0	0
186	0	0	0	0	0	6.2 11 19.8	0	0	0	0
187	0	0	0	0	0	0	6.2 11 18.9	0	0	0
188	0	0	0	0	0	0	6.2 11 18.1	0	0	0
189	0	0	0	0	0	0	6.2 11 17.3	0	0	0

Image Display Window 1

Figure 4-2:Data Values in the Boundary

Filler points are added to create a connected boundary, since the conversion from cylindrical coordinates in which the data is taken, to cartesian coordinates in which the data are displayed creates a sparse boundary. Every data point in the boundary carries with it information such as its index into the boundary, its x-y position, the angle from the robot at which the reading was taken with respect to a coordinate system attached to the robot, the radial distance measured by the sonar and the infrared reading. Figure 4-2 shows the actual data values along one small section of the boundary.

The circled region pointed to by the arrow corresponds to the partial array of numbers to the left. The zeroes represent white space while the other numbers represent either actual data points, filler points or room obstacles. For real data points, there is a set of three numbers: the radial distance in feet, the infrared reading, and the angle from the robot in degrees. For instance, the point at (x y) = (231 186) is 6.2 feet from the robot, has the infrared reading 11 associated with it, and lies at an angle of 19.8 degrees with respect to the robot. Filler points in the boundary are marked as ones and obstacles and walls are marked as sixes.

Examples of Sonar Errors

Many of the problems with the sonar sensor that were described in the previous chapter show up clearly in plots of the experimental data. Figure 4-1 depicts how doorways can be blurred so much that they look like walls. In addition, it also vividly displays the problem of the sonar measuring the shortest distance within its beam, and not necessarily the distance along the beam's centerline, as shown by the arc of data points along the lefthand wall. Where the wall is perpendicular to the beam, the distance measured is perfect, but to either side of that point, the return is shorter than it should be.

The sonar not only blurs out doors, but also makes small thin obstacles look very wide. Figure 4-3 shows a pole towards the lefthand side of the image which appears to be as wide as the sonar beam. Another problem occurs when the sonar is pointed towards a corner of the room. Either side of the corner is closer to the sensor, so an underestimate is always returned. Consequently, the real corners of the room become very hard to locate. This phenomenon appears in the upper righthand corner of Figure 4-4.

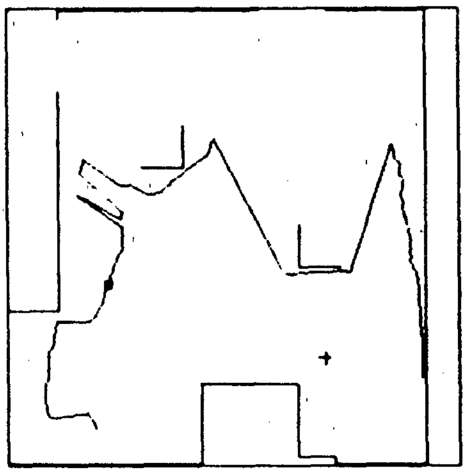


Figure 4-3:A Pole Creates a Wide Sonar Image

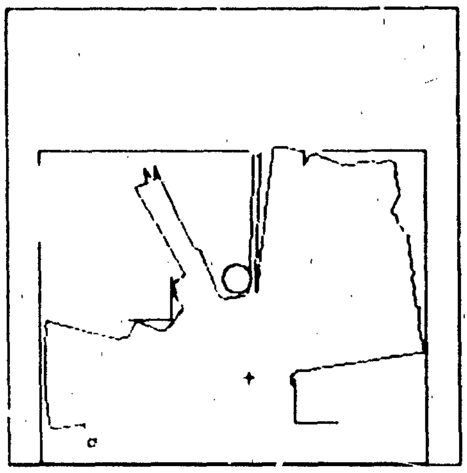


Figure 4-4:Corners Appear Closer Than Actual

A good example of the specular reflection problem is portrayed by Figure 4-5. Two L-shaped obstacles were placed in the room and the robot is about a foot and a half from one of them. The obstacle really was a folding closet door that had a series of horizontal slats that were angled downward. The angle of these surfaces caused the beam to bounce off in some other direction before finally being detected by the transducer, giving a distance reading much longer than was really the case. The result is that the robot is blind to

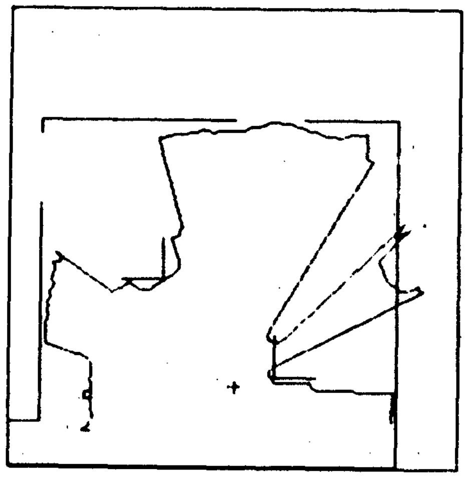


Figure 4-5:Specular Reflection Caused by Angled Surfaces

portions of the obstacle right in front of it.

Infrared Results

Under certain conditions, the infrared sensor is capable of locating edges of doorways very well. It does this by noticing changes from where it detects something to where it detects nothing. If there is a long clear distance beyond the door and the edge of the door is within the infrared's range, then that edge can be located with a very good angular resolution.

It was hoped that the stepped power output of the emitter and the four levels of sensitivity of the detector would enable the sensor to detect less drastic changes in depth, and therefore enable the sensor to discern if one object was a few feet in fron of another. It turned out however, that the sensor was not capable of such performance because the varying reflectivities of surface materials in the room precluded any attempt at correlating range with either emitter output or detector sensitivity. Nevertheless, the infrared sensor could consistently pick out edges of doorways that were completely invisible to the sonar.

Figure 4-6 shows two scenes in which the sonar blurs the doors, but the infrared is able to pick out the edges very accurately. The crosses mark where the infrared detects corners, or specifically, where infrared readings of 30 transitioned to anything else.

Readings of 30 indicate that all three LEDs were fired and no echo was detected. In the plot on the left, the sonar is able to see a passageway through the door at the top of the picture and also through the door on the left. However, the sonar smears the edges of the lefthand doorway, but the crosses marked by the infrared make it obvious that the infrared was able to find the doorway edges where they really were. Similarly for the plot on the right, the crosses at the top of the room show that the infrared correctly finds the edges of the dorrjam again, while the sonar is completely blind to them.

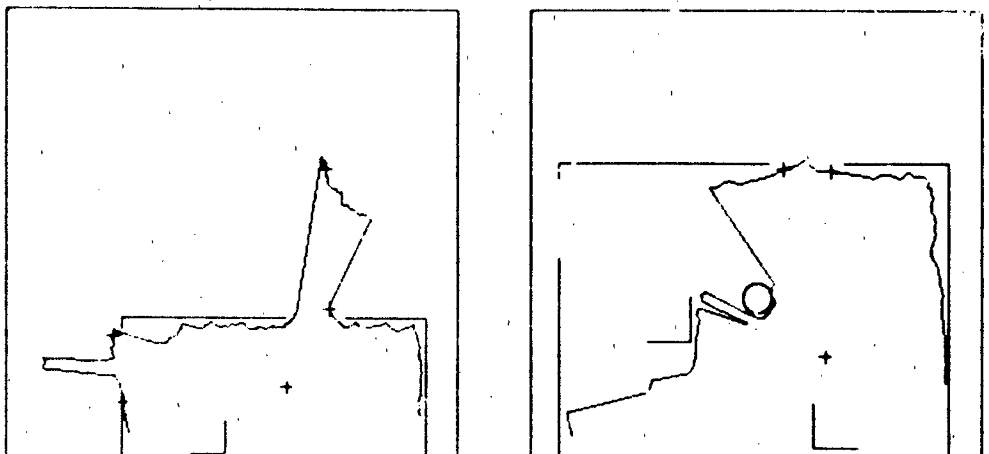
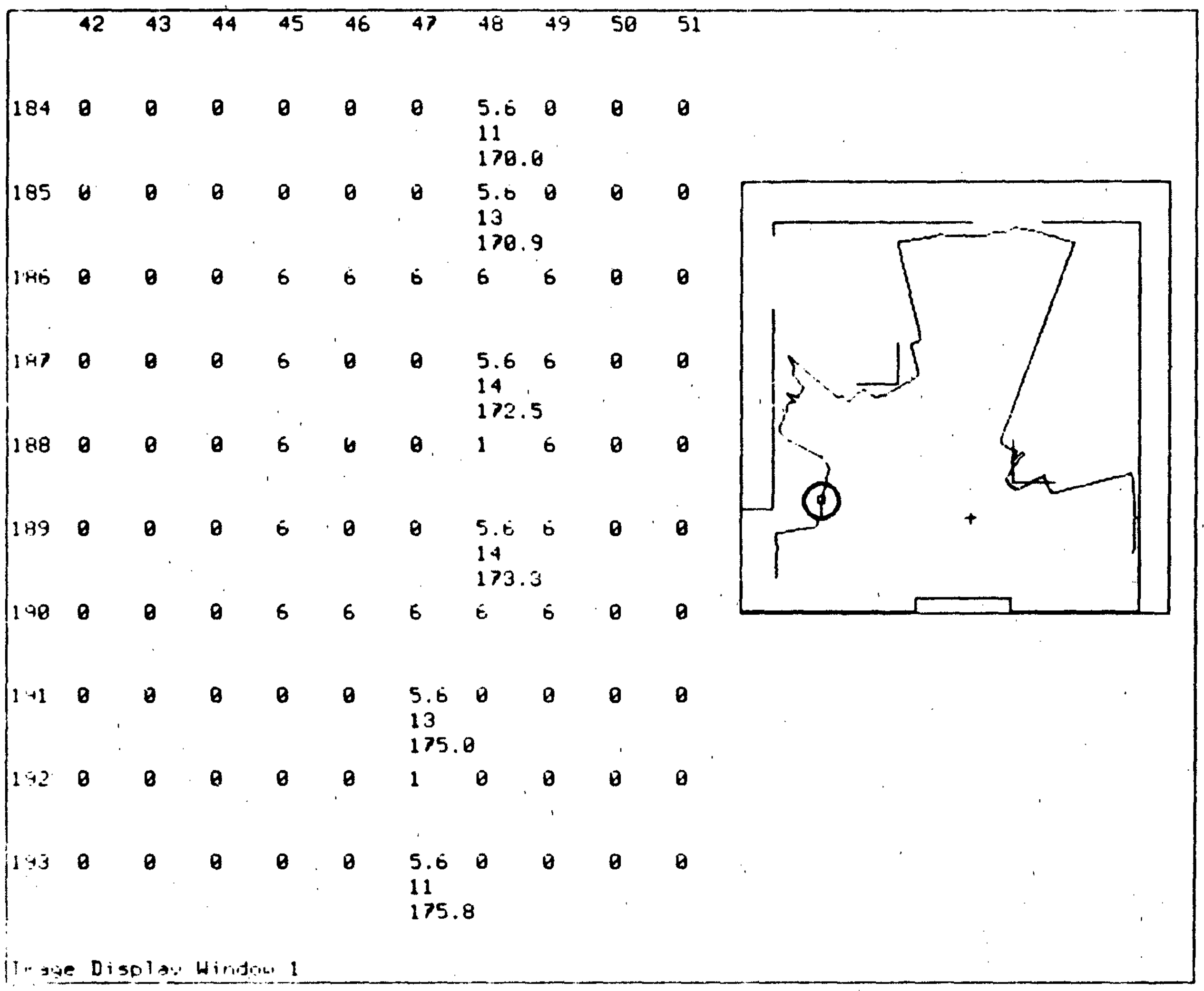


Figure 4-6: Infrared Sensor Picks Out the Doors

However, the infrared sensor isn't so good at finding edges of obstacles that don't have long empty region of space behind them. Figure 4-7 shows a pole blurred by the sonar. The sixes in the array to the left mark the location of the pole, but the infrared values nearby don't show much change. The values starting with the point (48 184) and following the edge of data points down to point (47 193) are the second items in each data point (12, 13, 11, 13, 14, 14, from top to bottom). This is because the wall behind the pole is relatively close.

The problem that arises in trying to correlate range with emitter output can be see in Figure 4-8. The points (174 79) and (175 79) show their infrared values swinging from 31 to 11, yet there is no large jump in distance. The edge of the door has been detected between

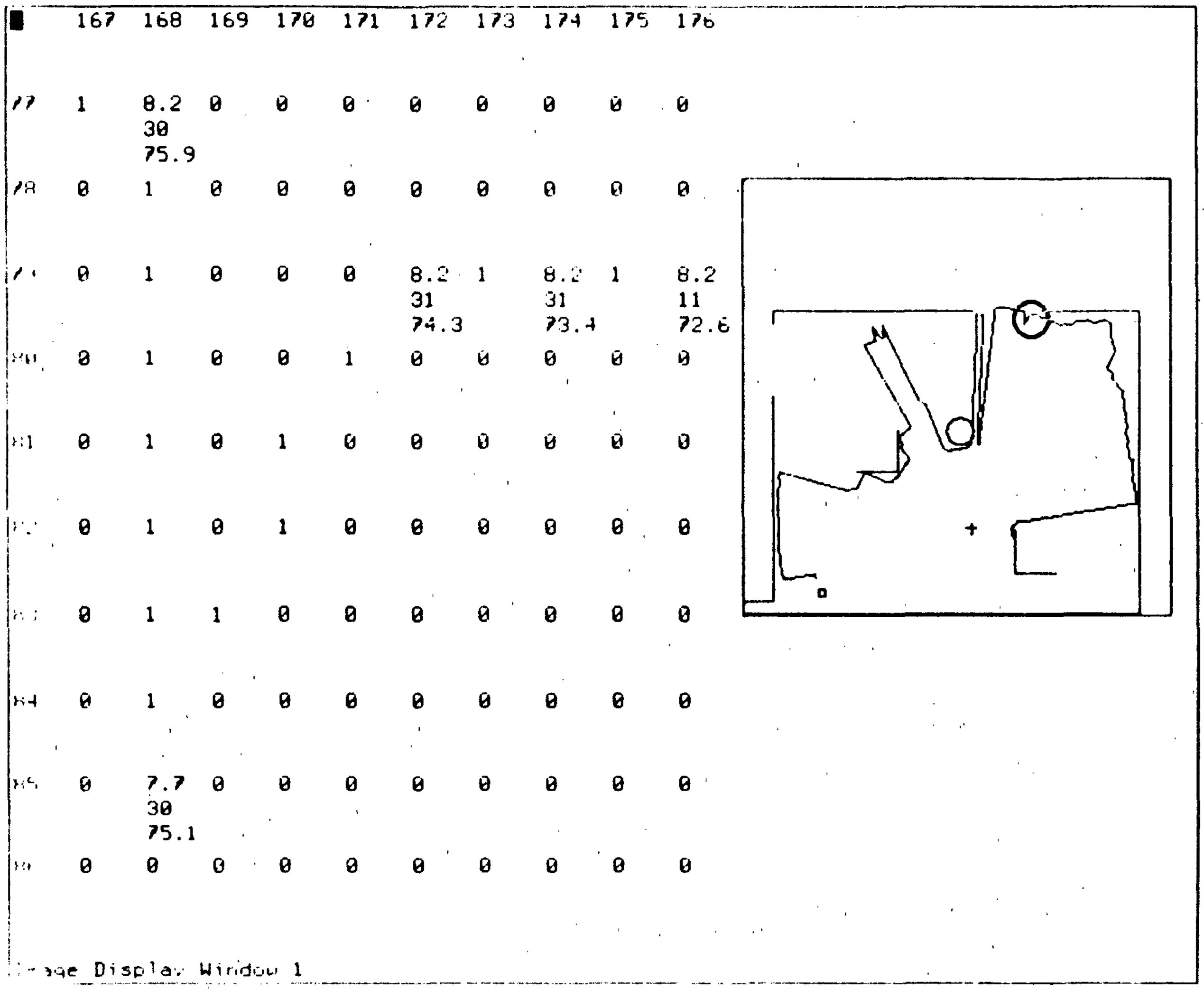


```
     42   43   44   45   46   47   48    49   50   51

184  0    0    0    0    0    0    5.6   0    0    0
                                  11
                                  170.0
185  0    0    0    0    0    0    5.6   0    0    0
                                  13
                                  170.9
186  0    0    0    6    6    6    6     6    0    0


187  0    0    0    6    0    0    5.6   6    0    0
                                  14
                                  172.5
188  0    0    0    6    6    0    1     6    0    0


189  0    0    0    6    0    0    5.6   6    0    0
                                  14
                                  173.3
190  0    0    0    6    6    6    6     6    0    0


191  0    0    0    0    0    5.6  0     0    0    0
                             13
                             175.0
192  0    0    0    0    0    1    0     0    0    0


193  0    0    0    0    0    5.6  0     0    0    0
                             11
                             175.8

Image Display Window 1
```

Figure 4-7:Infrared Can't Find the Pole

the 30 and 31 transition, the points (168 77) and (172 79).

The point which has the infrared reading of 11, point (176 79), illustrates the crux of the problem. It is actually 8.2 feet away, so the ranges available with the various stepped outputs of the emitter overlap quite a bit. This means that the infrareds can't really be used to signal a specular reflection caused by the sonar either. Take for example, Figure 4-9, which clearly shows the sonar wrongly stating that there aren't any obstacles for seven feet.

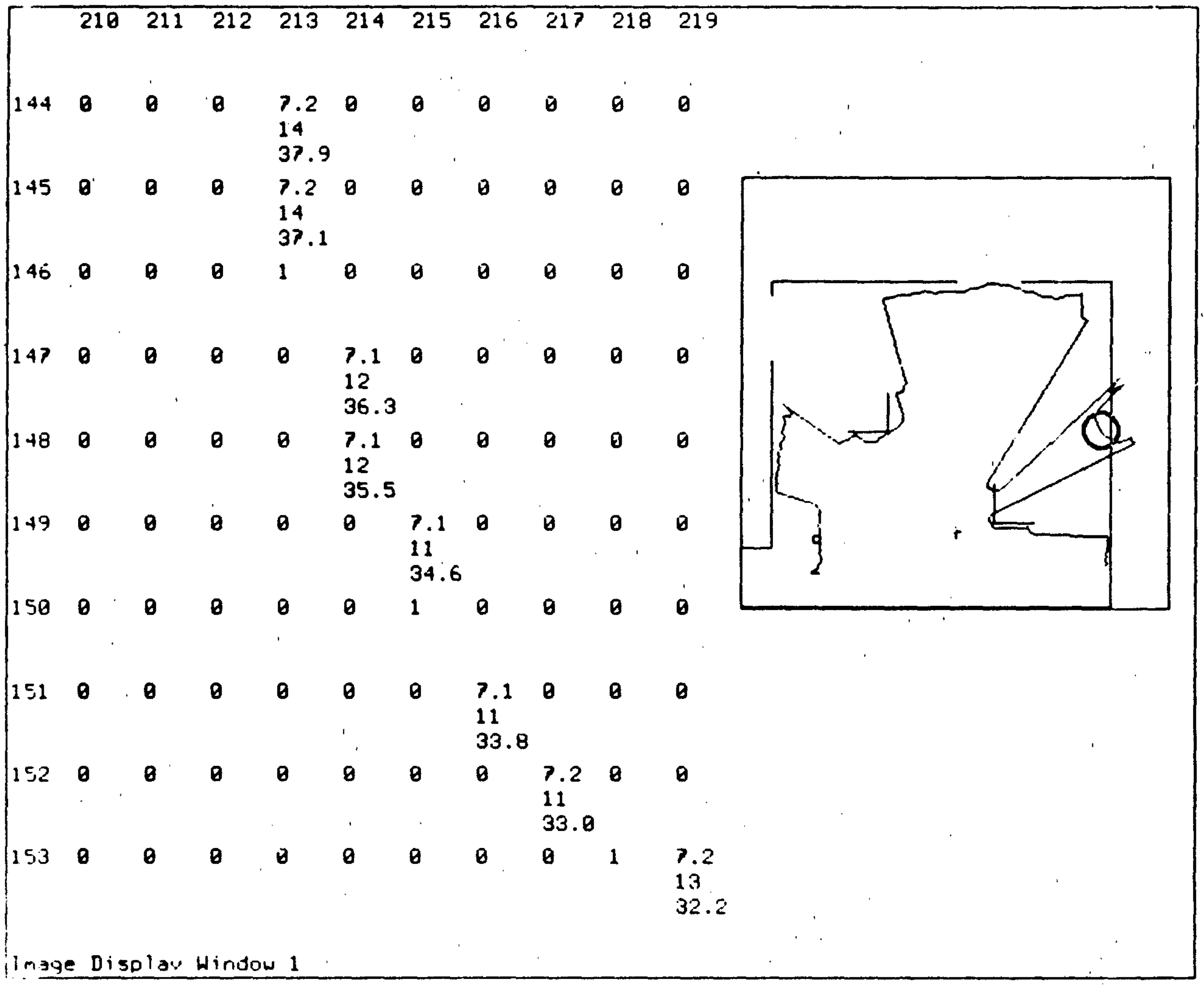


```
█    167   168   169   170   171   172   173   174   175   176

77   1     8.2   0     0     0     0     0     0     0     0
           30
           75.9
78   0     1     0     0     0     0     0     0     0     0

79   0     1     0     0     0     8.2   1     8.2   1     8.2
                                   31          31          11
                                   74.3        73.4        72.6
80   0     1     0     0     1     0     0     0     0     0

81   0     1     0     1     0     0     0     0     0     0

82   0     1     0     1     0     0     0     0     0     0

83   0     1     1     0     0     0     0     0     0     0

84   0     1     0     0     0     0     0     0     0     0

85   0     7.7   0     0     0     0     0     0     0     0
           30
           75.1
86   0     0     0     0     0     0     0     0     0     0

Image Display Window 1
```

Figure 4-8: Infrared Ranging Is Hazy

Even though the infrared readings appear to detect a close-by object, it can't overrule the sonar because in some cases one LED can see out to 8.2 feet as in the example above.

In other cases, the infrared sensor detects edges where there are none. This happens when the sensor is scanning along a wall and reaches the horizon of its range. It suddenly notices a change from seeing something to seeing nothing, not because it has found a doorway or a corner, but because the wall is going in such a direction that it veers out of the



	210	211	212	213	214	215	216	217	218	219
144	0	0	0	7.2 14 37.9	0	0	0	0	0	0
145	0	0	0	7.2 14 37.1	0	0	0	0	0	0
146	0	0	0	1	0	0	0	0	0	0
147	0	0	0	0	7.1 12 36.3	0	0	0	0	0
148	0	0	0	0	7.1 12 35.5	0	0	0	0	0
149	0	0	0	0	0	7.1 11 34.6	0	0	0	0
150	0	0	0	0	0	1	0	0	0	0
151	0	0	0	0	0	0	7.1 11 33.8	0	0	0
152	0	0	0	0	0	0	0	7.2 11 33.0	0	0
153	0	0	0	0	0	0	0	0	1	7.2 13 32.2

Image Display Window 1

Figure 4-9:Infrared Can't Adjust For Sonar's Specular Reflection

limits of the sensor. The crosses in Figure 4-10 show where the infrared just starts to detect the wall.

There are yet other problems. Since the detector is focused over a very narrow area, thin objects can often be overlooked. This is one problem the sonar has no trouble with, however. In Figure 4-11, the infrared fails to detect the pole at a distance of 8.2 feet, yet in Figure 4-12, it clearly finds a wall at 10 feet. The parabolic reflector which limits the area

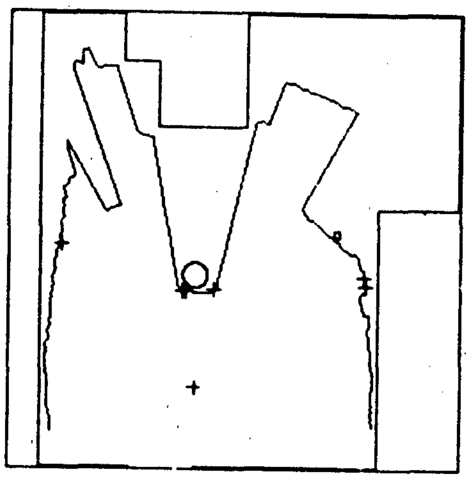


Figure 4-10: Infrared Reaches the Limits of Its Range

over which the infrared detects echoes, also causes it to miss relatively thin obstacles during a scan.

Rules for Fusing Information

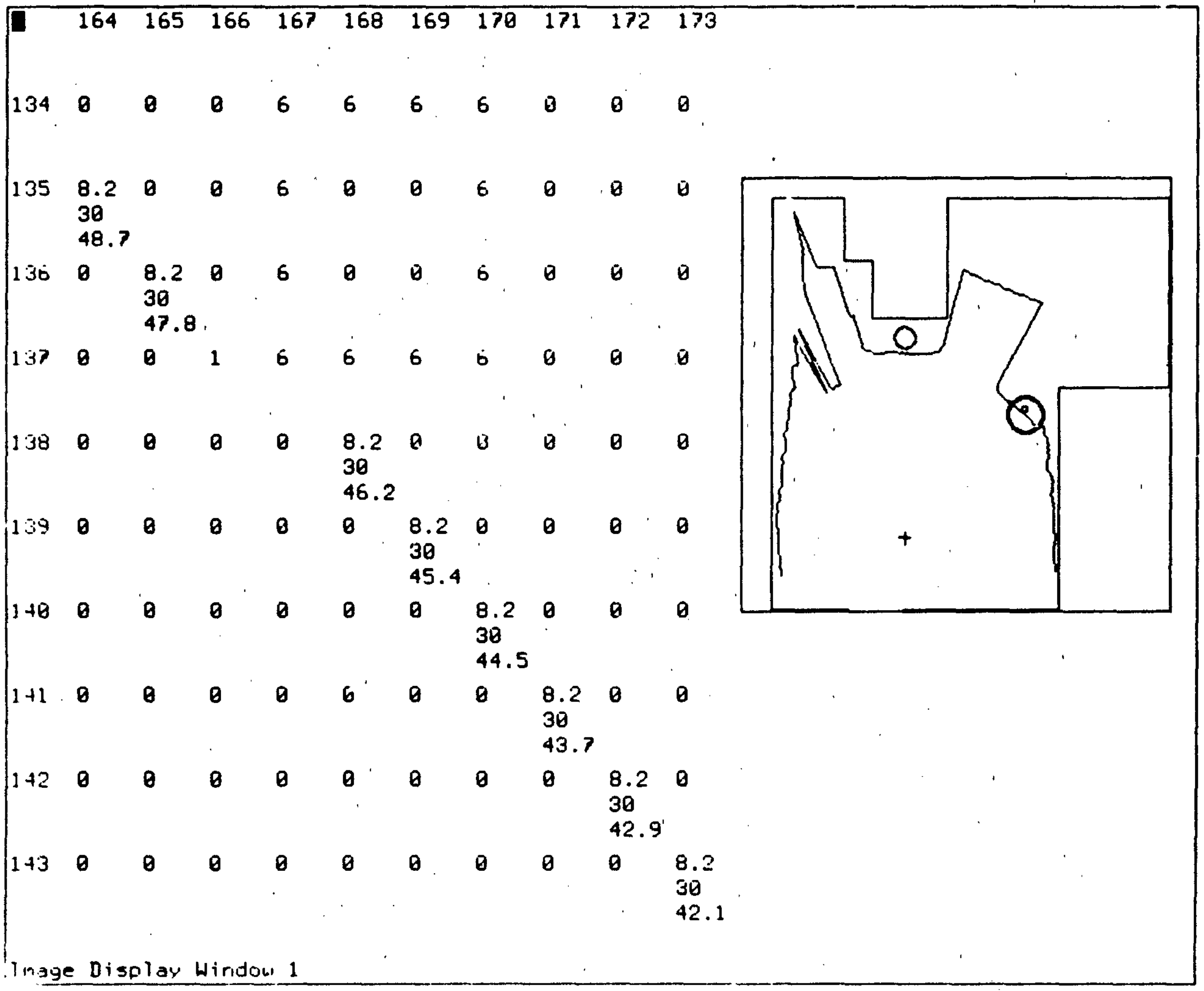
Taking into account all the limitations listed above, the following rules are used for fusing information from the two sensors:

* Whenever the infrared sensor detects a 30-to-other discontinuity, a change from detection to no detection, and the associated sonar reading is less than 10 feet, then it's very likely that a valid depth discontinuity has been detected.

* If the sonar reading is greater than the maximum range for the infrared, then ignore the infrared.

* If the sonar reading is at its maximum value, then the real distance is greater.

Using these few rules the original sonar boundary can be redrawn to take into account any passageways found by the infrared sensor. This is done by finding a pair of edges and redrawing the boundary in between to be an arc at the maximum of either the horizon of the infrared or the furthest sonar reading returned between the two edges of the passageway. Figure 4-13 shows an original boundary and the redrawn map based on information from both sensors. The original boundary is shown in the right hand picture with the new boundary overlaid on it at the locations where the infrared marked the edges.



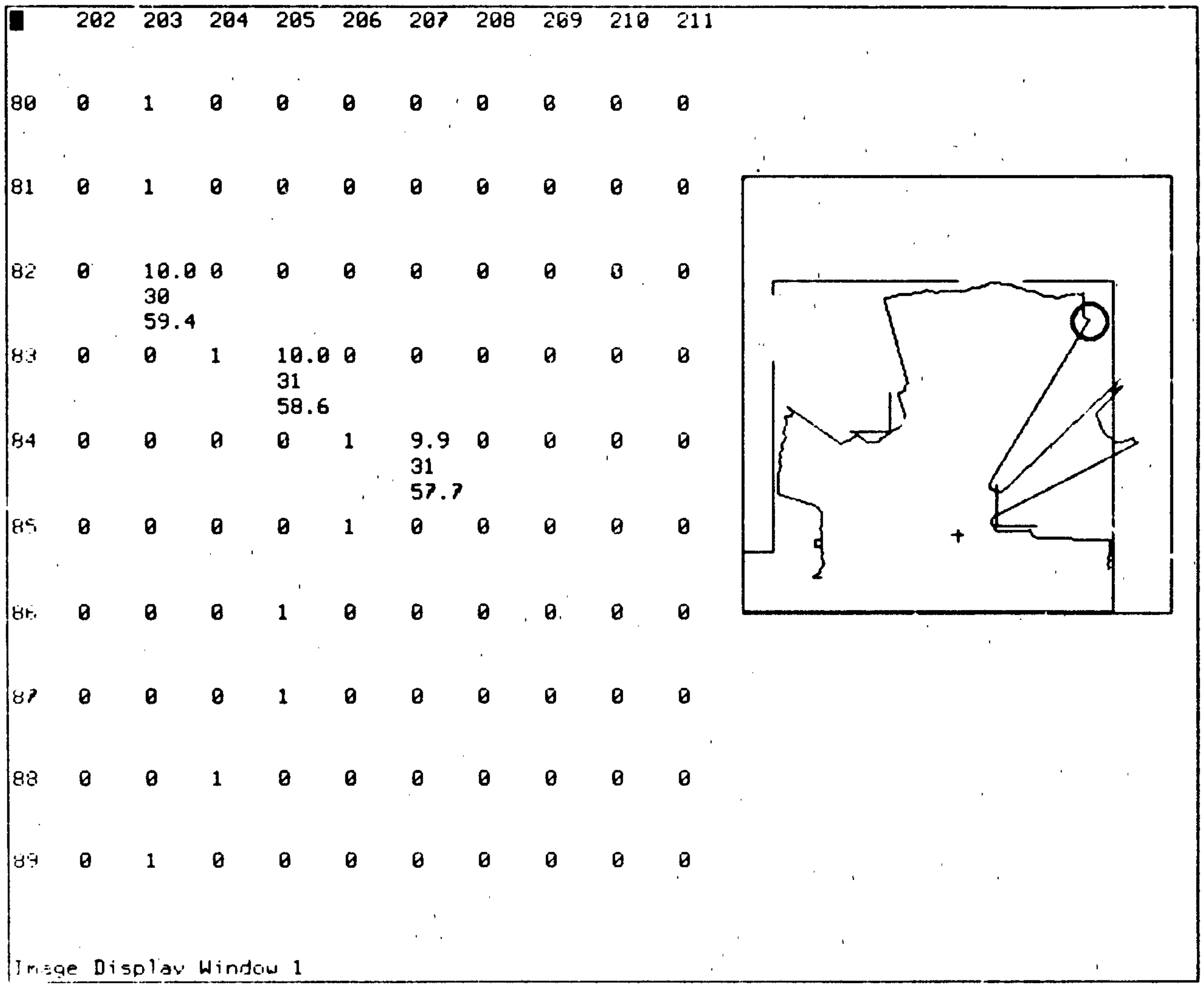
	164	165	166	167	168	169	170	171	172	173
134	0	0	0	6	6	6	6	0	0	0
135	8.2 30 48.7	0	0	6	0	0	6	0	0	0
136	0	8.2 30 47.8	0	6	0	0	6	0	0	0
137	0	0	1	6	6	6	6	0	0	0
138	0	0	0	0	8.2 30 46.2	0	0	0	0	0
139	0	0	0	0	0	8.2 30 45.4	0	0	0	0
140	0	0	0	0	0	0	8.2 30 44.5	0	0	0
141	0	0	0	0	6	0	0	8.2 30 43.7	0	0
142	0	0	0	0	0	0	0	0	8.2 30 42.9	0
143	0	0	0	0	0	0	0	0	0	8.2 30 42.1

Image Display Window 1

Figure 4-11:The Infrared Misses the Pole at 8.2 Feet

The infrared edges are marked by the crosses, and those are the positions at which the infrared noticed a transition from detecting something to detecting nothing (and vice versa for the other edge). The left hand figure shows the modified boundary after combining the infrared data with the sonar. It's clear that the doors are more pronounced after filtering with the infrared.

Due to the problems mentioned earlier about the infrared falsely detecting doors,



```
█      202     203     204     205     206     207     208     209     210     211

80     0       1       0       0       0       0       0       0       0       0

81     0       1       0       0       0       0       0       0       0       0

82     0       10.0    0       0       0       0       0       0       0       0
               30
               59.4
83     0       0       1       10.0    0       0       0       0       0       0
                               31
                               58.6
84     0       0       0       0       1       9.9     0       0       0       0
                                               31
                                               57.7
85     0       0       0       0       1       0       0       0       0       0

86     0       0       0       1       0       0       0       0       0       0

87     0       0       0       1       0       0       0       0       0       0

88     0       0       1       0       0       0       0       0       0       0

89     0       1       0       0       0       0       0       0       0       0

Image Display Window 1
```

Figure 4-12:The Wall is Detected at 10 Feet

because it has reached the limits of its range, sometimes the boundaries are redrawn incorrectly as in Figure 4-14. The real distance to the wall is beyond the horizon of the infrared, but the sonar returns a foreshortened reading because it's aimed towards a corner. Consequently, the infrared data is not ignored, and the robot thinks it has found another door. However, after a sequence of moves the robot will see the same walls from different vantage points and these false doors can be dismissed. This modified boundary is then

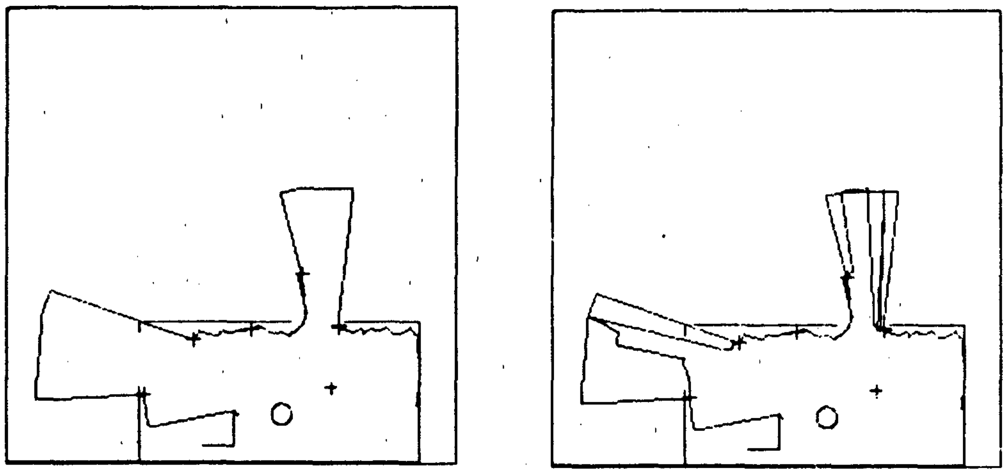


Figure 4-13:Redrawn Map From Combining Sensors

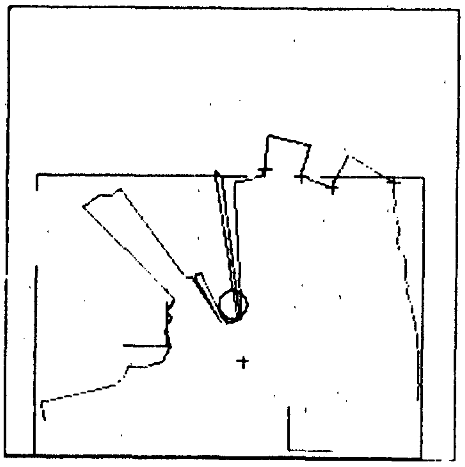


Figure 4-14:Infrared Detects False Door in the Corner

input to the next stage of processing which builds a representation of the room better suited for a path planner.

Chapter Five

Building a Representation

After combining data from two sensors to create a refined view of the room, this new map needs to be converted to a representation suitable for planning intelligent tasks, such as navigating through a workspace. This is achieved by first transforming the refined data to an intermediate representation, the curvature primal sketch [Brady 84], which is convenient for merging separate views between robot moves. The curvature primal sketch representation can then be easily converted into a polygonal representation of the world suitable for path planners [Brooks 83, Brooks 85].

The curvature primal sketch describes a boundary by a spline whose knot points are significant changes in curvature. These knot points are located by taking first and second derivatives of Gaussians at various scales which have been convolved with the boundary, and then looking for patterns of zero crossings, maxima and minima of the smoothed result that correspond to models of curvature change such as those produced by corners, ends and smooth joins. Figure 5-1 shows an example of the curvature primal sketch at two scales along with the refined boundary. Instead of fitting a spline to these knot points, however, straight line approximations are used. Straight line approximations are appropriate for a robot's world in which walls, desks, tables and other obstacles are most often composed of straight edges.

There are some nice features of the curvature primal sketch which make it a desirable representation for this application, the task of building a room map out of several views from different positions as the robot goes exploring. First, the knot points are convenient for matching subsequent scans because they have local support. That is, the curvature primal sketch isn't based on any global properties of the data plot of the room, such as length, width, or aspect ratio. Rather, the knot points are determined only by their relationship to their neighbors. This is important when trying to merge several views of the

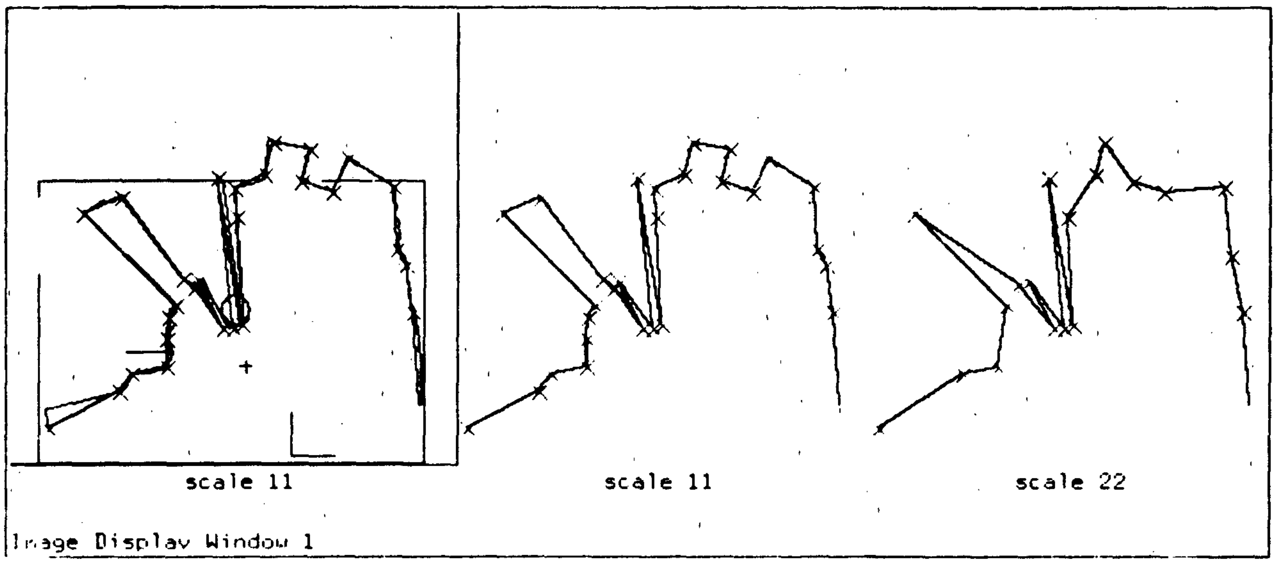


Figure 5-1:The Curvature Primal Sketch of the Refined Boundary

room, because from different viewpoints the global shape of the room can change drastically. Some objects can become occluded while others become uncovered. The knot points in the curvaure primal sketch, however, will stay close to the same location for incremental changes in the robot position. Figure 5-2 shows the curvature primal sketch of the robot's view after it has moved two feet to the right with respect to Figure 5-1. From its new position, the robot is unable to see the same areas behind the circular object as from its earlier position. Figure 5-3 shows the robot's perspective after it moves two feet forward from its position in Figure 5-2. Previously occluded areas can be seen to become uncovered. However, edges that were visible from both perspectives have similar knot points.

Second, the knot points are found reliably and consistently due to processing at multiple scales. Gaussian filters with a large base of support smooth out noise and detect occurences of curvature changes, while gaussians with small support can then be used to localize those occurences. This is analogous to previous work on finding edges in images [Canny 83].

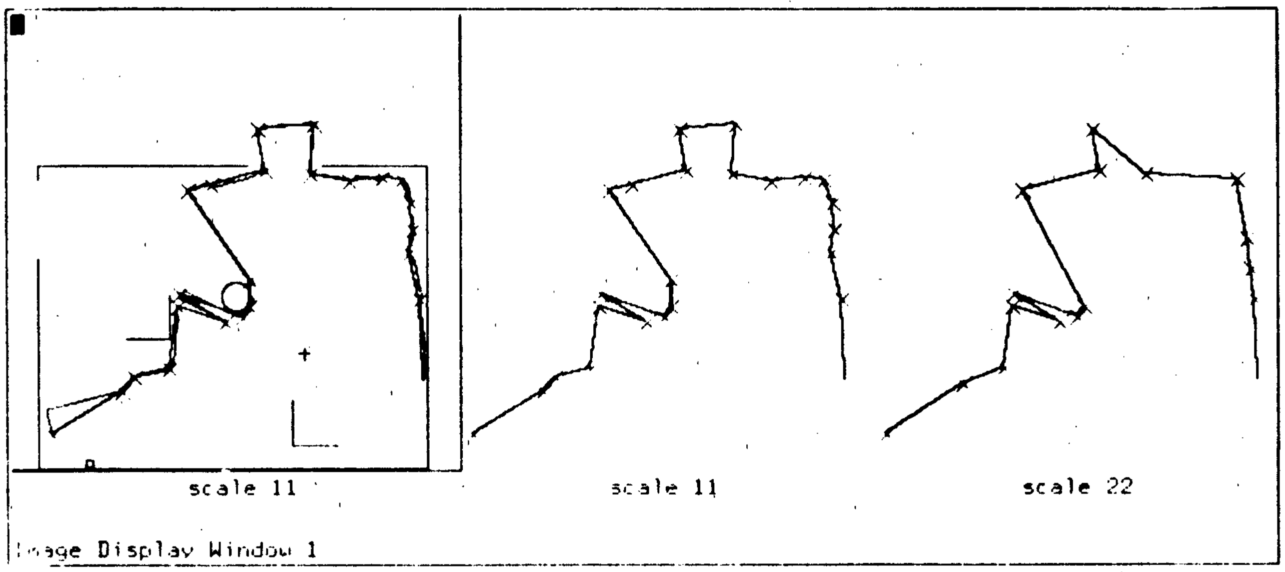


Figure 5-2: The Robot Moves Two Feet To The Right

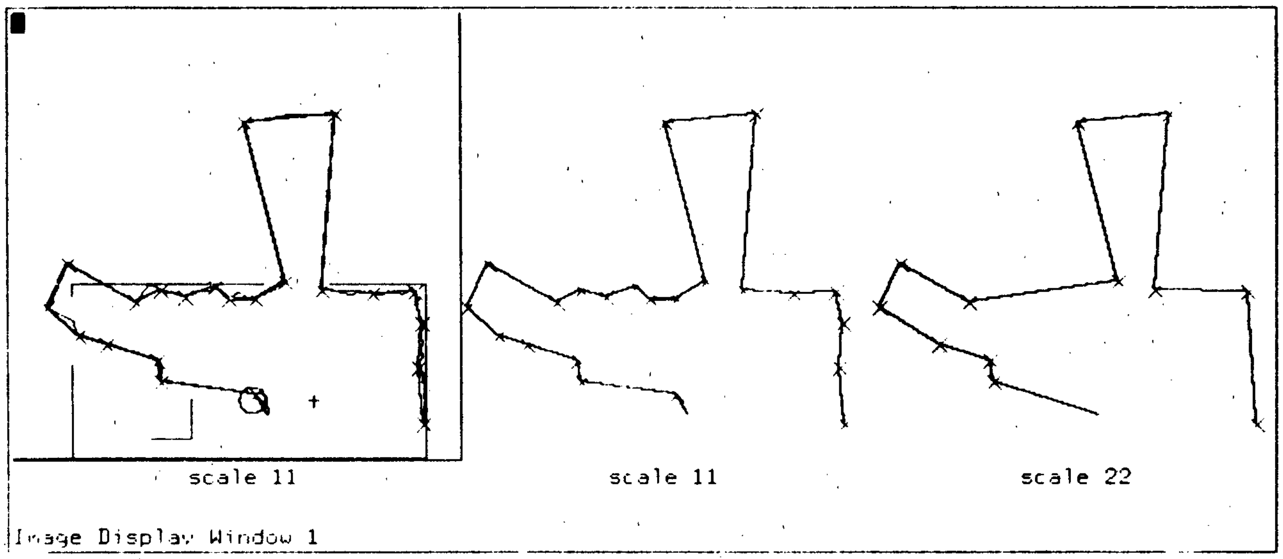


Figure 5-3: The Robot Then Moves Two Feet Forward

Finally, the curvature primal sketch provides a more concise representation than the raw data and acts as an important intermediate representation before converting to one more suitable for planners. Path planning programs typically expect a list of polygons as input for their simulated worlds. The straight line segments between knot points can easily be converted to very thin rectangles, translated and rotated appropriately.

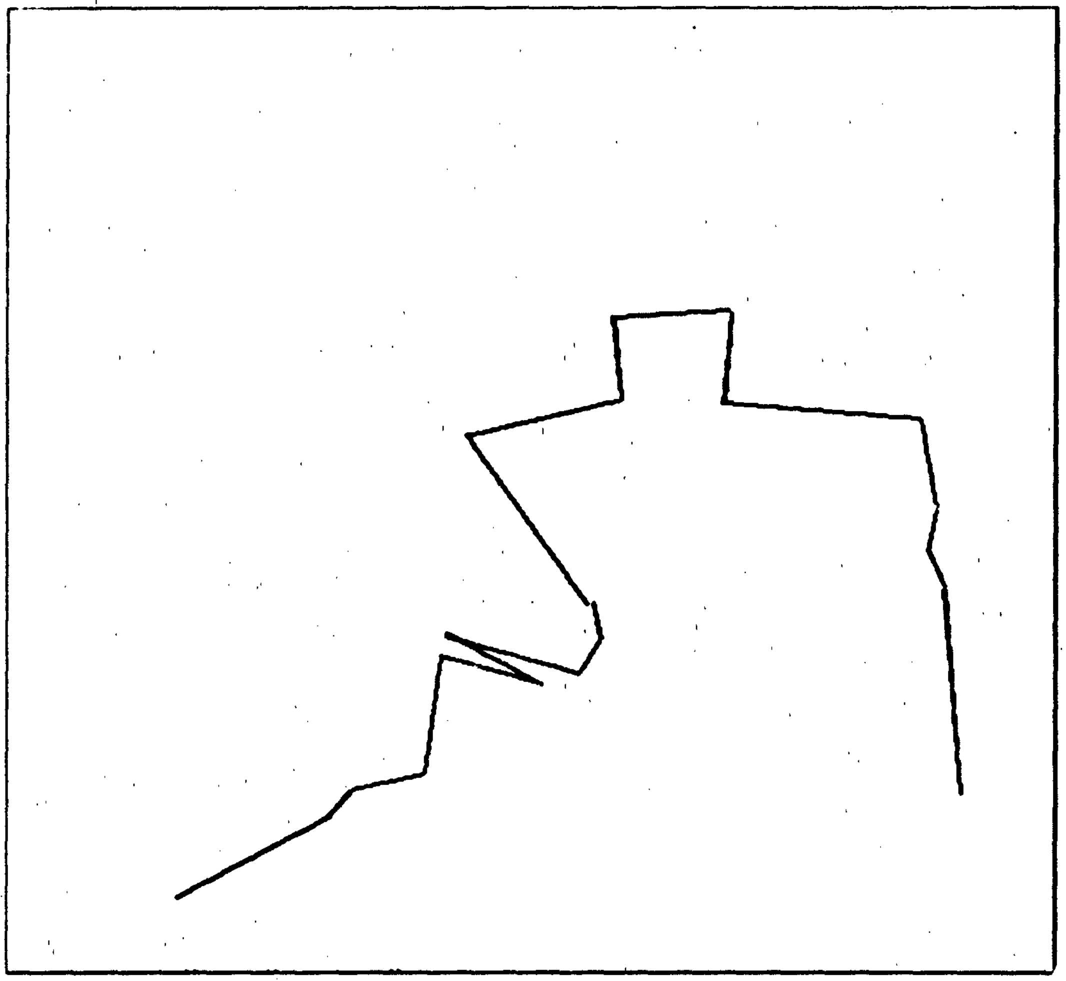


Figure 5-4:Polygonal Representation for a Planner

This conversion is shown in Figure 5-4 where the plot of Figure 5-2 is represented as a set of linked rectangles of width one, ready for input to a planner based on generalized cones [Brooks 83]. This planner finds freeways and channels in the room by carving up the freespace into generalized cones. Generalized cones have a spine, and are parameterized by some sweeping rule which says how the width to either side of the cone varies. All pairs of edges of the polygons in the workspace are compared and heuristics are used to prune the number of cones generated. The spines of the resulting cones are illustrated in Figure 5-5.

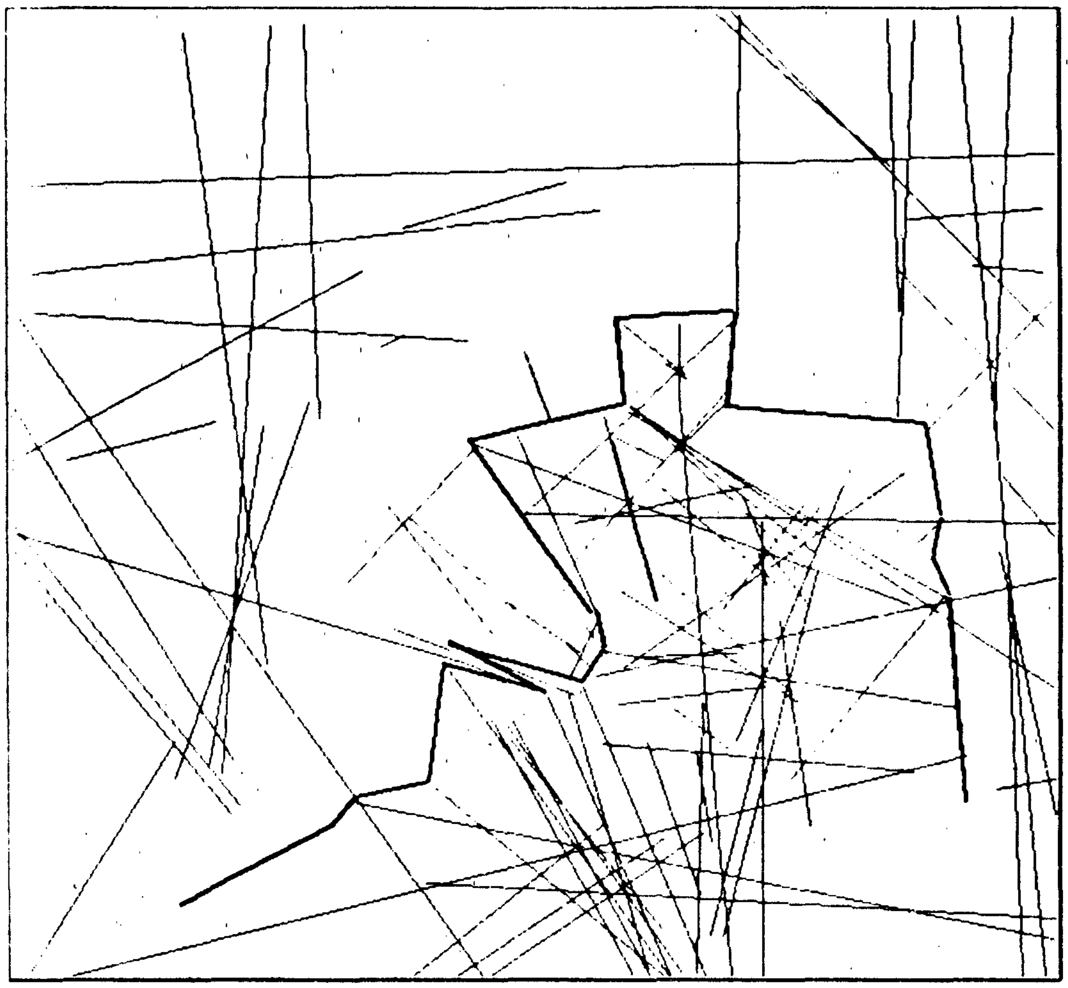


Figure 5-5:Spines of Generalized Cones Representing the Freespace

Paths from start to goal are determined by searching the spines of the cones for the optimal route. The robot translates along the length of the spines and rotates at the intersection of two spines. The path found from specified start and goal locations is shown in Figure 5-6. This path, slightly more complicated than need be, shows that this planner doesn't work well with lots of obstacles. This is also clear from the number of cones generated in Figure 5-5.

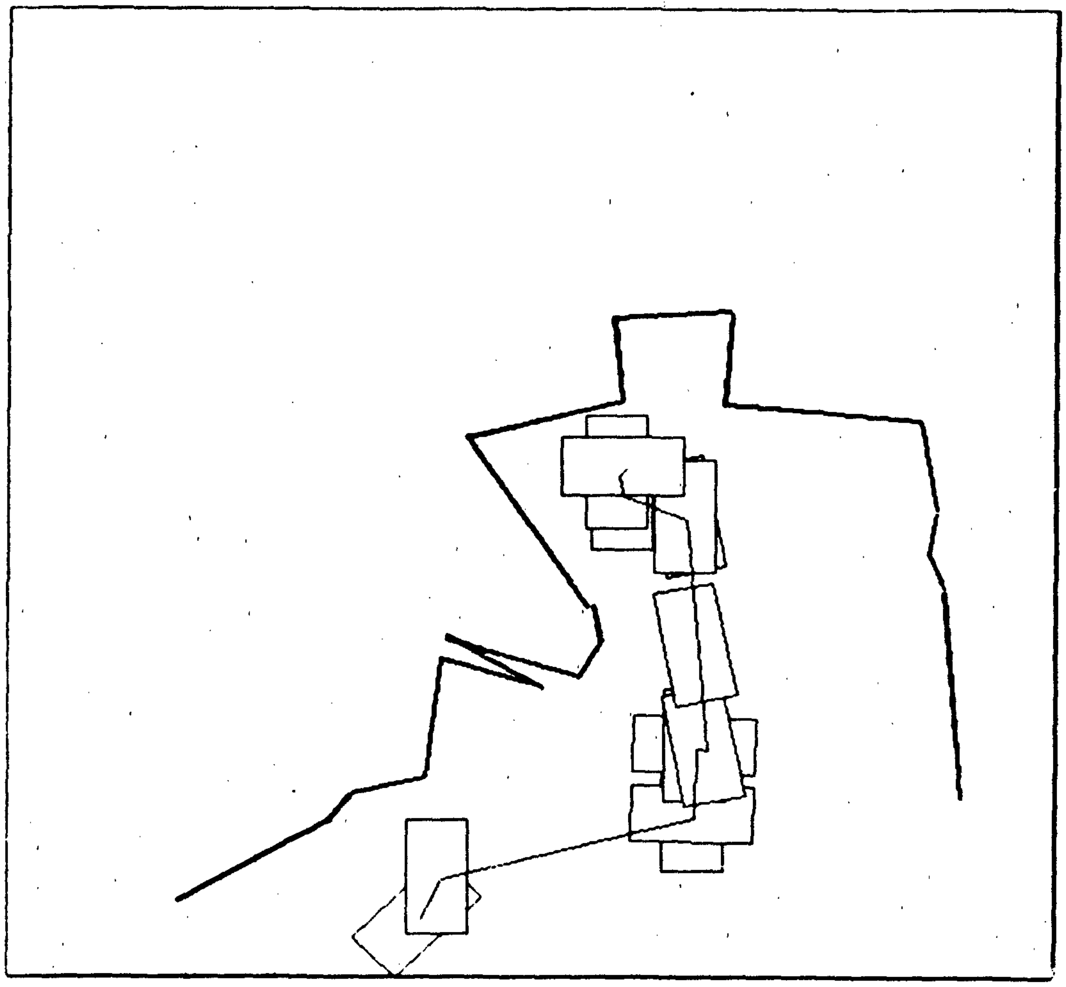


Figure 5-6:A Found Path

All of this has been an example of how raw data taken from one scan of a room has been refined and converted into the appropriate representation for a planner. However, what is really desired, is a global map built by merging several scans of the room taken from different locations. Problems in doing this arise from having uncertainties both in the sensed data and in the distance and direction moved by the robot. In trying to merge several views from different perspectives, the robot has to be intelligent about what new pieces of data to include in his global model and what old pieces to throw away. Basically, it must match what he can and make decisions about what to do with the rest.

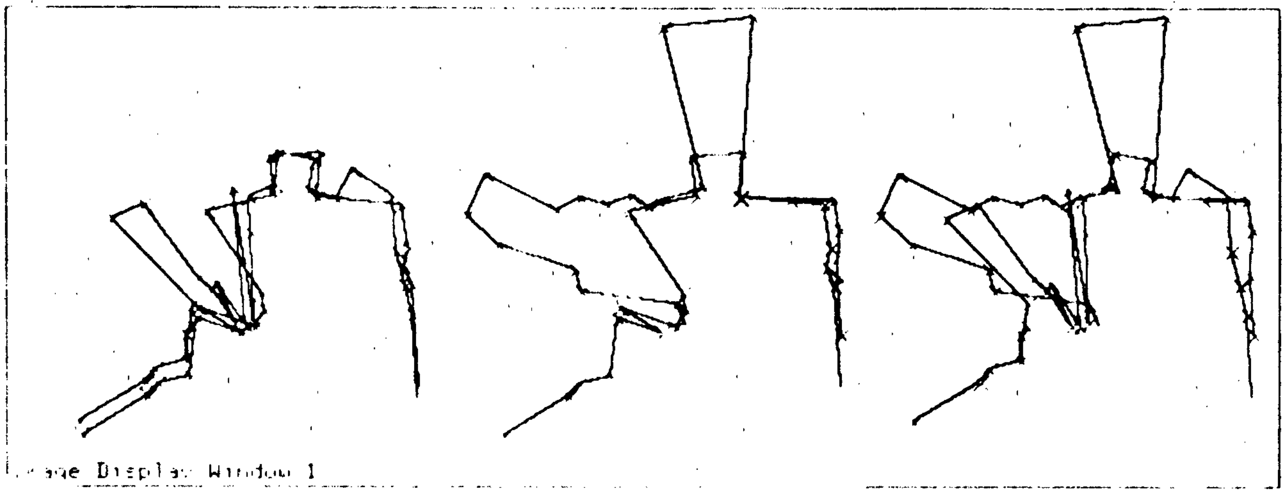


Figure 5-7:Three Views Overlaid, Two at a Time

This problem is illustrated in Figure 5-7. The earlier examples of three different views of the room are translated and scaled appropriately, then overlaid two at a time so as not to become too cluttered. It's clear that some edges, such as the wall to the right, tend to match up between views but that other edges, such as those behind the circular obstacle, tend to change. From the first position, the robot sees an area behind and to the left of the circular obstacle. From the second position, that area becomes occluded. After the robot moves forward, however, it again sees the same area behind the obstacle, only now it sees it from the left side. The robot should infer that these areas are connected and that a small object lies in the middle of the room. The desired output of a program that was intelligent about how to do such a merge might look something like Figure 5-8.

Some ideas on solving this problem [Chatila 85] were mentioned earlier in the section on Hilare. In that work, the robot scans the room from one perspective and assigns local

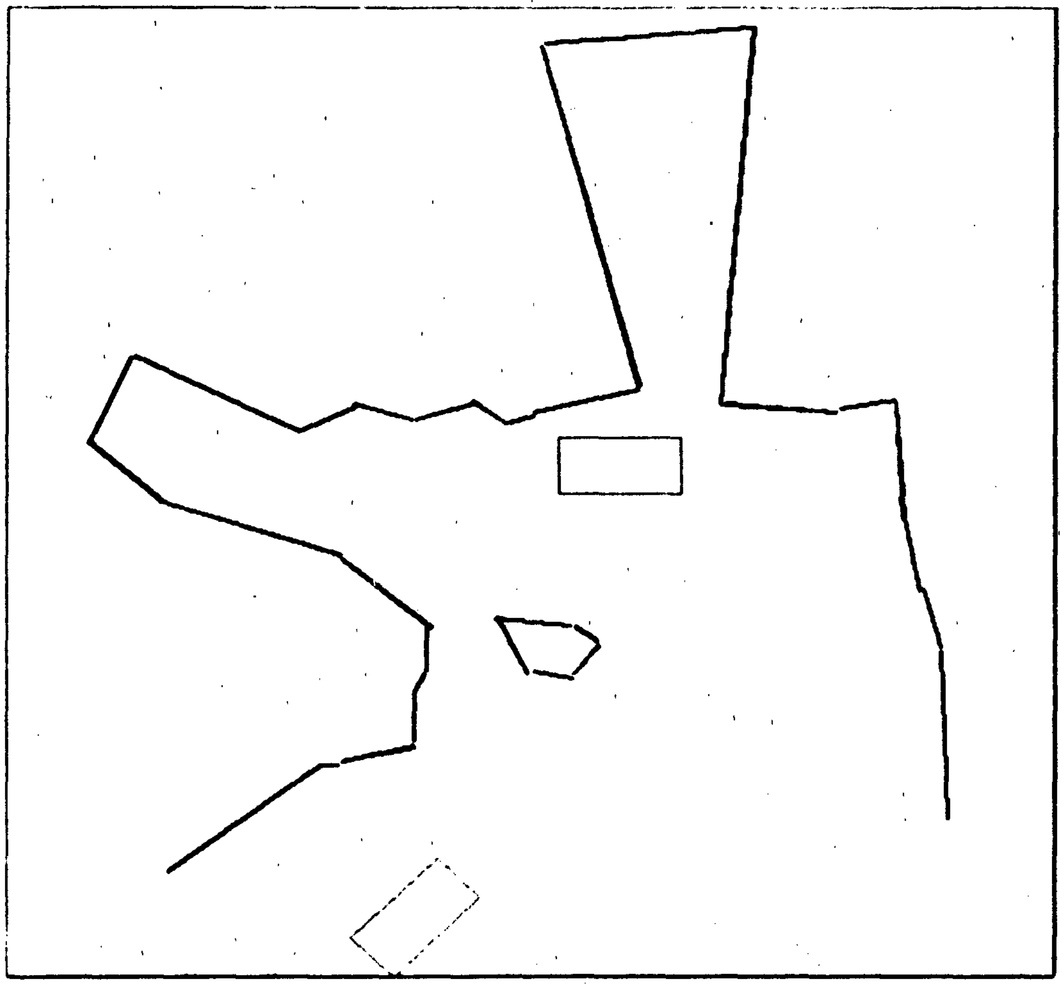


Figure 5-8: Desired Room Map After Merging Views

coordinate frames to clusters of data points that are line-fitted. Edges adjoining these local frames are marked as fake edges. By knowing which edges are fake and which are real, the robot can hypothesize about what it might see after its next move. That is, it can assume that new objects might become uncovered through these fake edges. It can also predict what objects might become occluded. Then after the next move and scan, it matches new edges to appropriate local coordinate frames of previously discovered objects. Having edges referenced to local coordinate frames reduces uncertainty in the building of the final map. For edges that can't be matched, decisions are made based upon whether or not the new information coincides with what was predicted.

Some of these ideas, modified slightly, can be incorporated into our problem of merging sonar maps. One big difference lies in the problem of deciding what to match. In the Hilare work, they assume they'll have laser rangefinder data which is much more accurate than sonar data. Consequently, the local coordinate frames can't be built here because the sonar blurs corners that come out towards the robot. However, the important idea of matching landmarks that have local support can be retained, because of the availability of the knot points from the curvature primal sketch. These knot points mark features in the scan that depend only on characteristics of neighboring points, namely curvature changes.

Furthermore, the fake edges described in the Hilare work are already included in the curvature primal sketch representation. These edges are made up of the filler points mentioned earlier, when the connected boundary was created from the sparse sonar data. The connected boundary was created from the raw data in order to make the curvature primal sketch code run, but explicitly keeping this information about filler points can be useful for marking fake edges between knot points. Knowledge about which edges between knot points are fake and which are real is useful for hypothesizing where obstacles can become occluded or uncovered.

After deciding what are close matches and what can be thrown away or added, algorithms appropriately unioning and intersecting the appropriate polygons can be taken from work in computational geometry [Weiler 77]. Basically, a union of the plots shown in Figure 5-7 would produce the desired output of Figure 5-8. The difficulty lies in making decisions about what constitutes a match and about what polygons to intersect or union when there isn't a match.

After a global room map is created, it can then be used as a model of the robot's entire workspace upon which it can plan tasks. An example of another planner, based on configuration space [Brooks 85], is shown in Figure 5-9. Configuration Space is a representation of the workspace in which the robot is shrunk to a point and the obstacles are

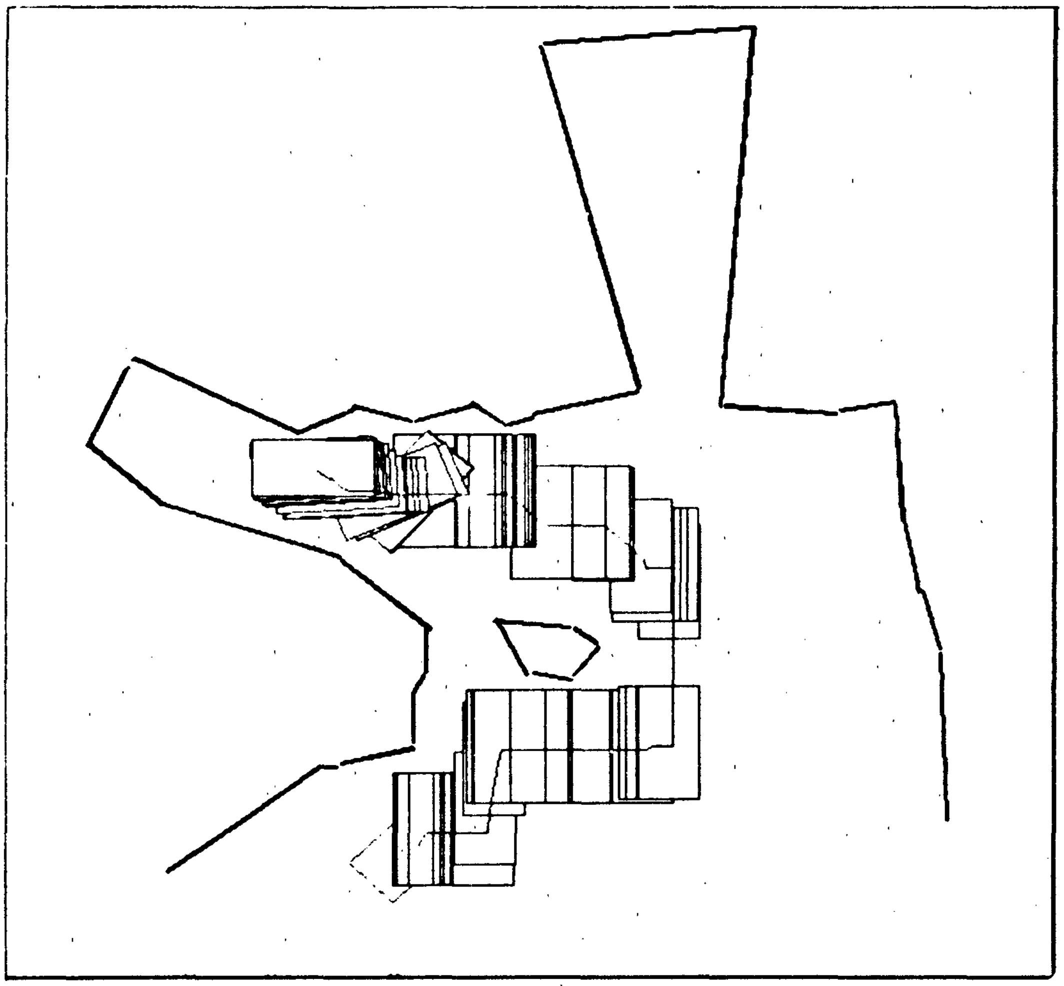


Figure 5-9: Configuration Space Planner

grown appropriately. Growing an obstacle by translating the robot to each of its vertices produces a two dimensional configuration space, and the problem of finding a free path for a polygonal robot through polygonal obstacles reduces to the problem of finding a path for a point through these grown polygons. However, allowing the robot to rotate, creates a three dimensional configuration space where the obstacles' surfaces can be curved, so the problem of finding the path of a point through this space becomes much harder. The solution this planner uses is to slice up the three dimensional configuration space and look for subpaths that involve only translation. It finds a path as far as it can, and then tries a new orientation in a different slice of the configuration space. This planner is guaranteed to find a path is one exists. A harder problem given to this planner is shown in Figure 5-10.

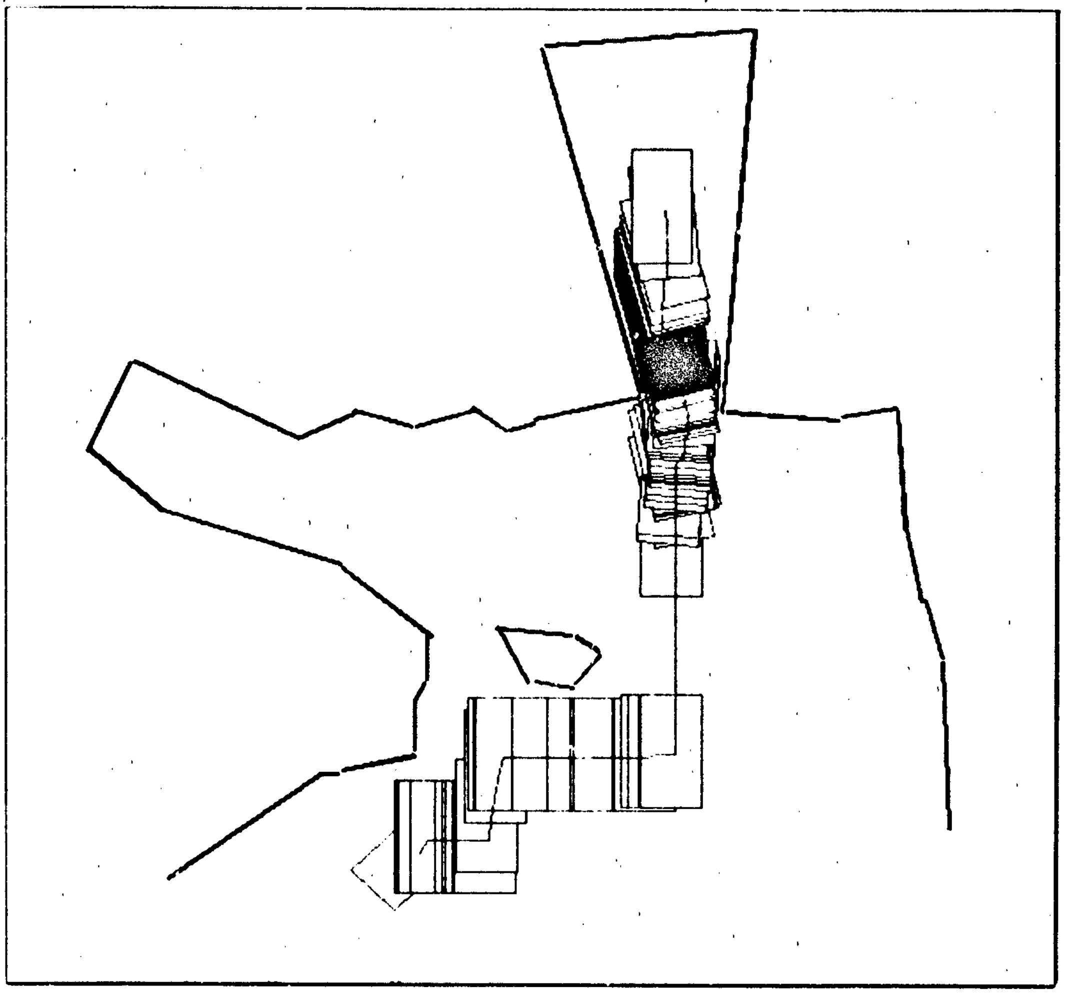


Figure 5-10:Harder Problem For the Planner

For a three dimensional world, configuration space gets very complicated. Allowing for three degrees of rotations produces a six dimensional configuration space. However, for mobile robots that don't fly, and for a two dimensional model of the world, a three dimensional implementation of configuration space works quite well.

Chapter Six

Conclusion

Two inexpensive sensors,a Polaroid sonar transducer and a novel infrared sensor have been combined to produce a refined map of the robot's workplace which is suitable for use by intelligent path planning programs.

The sensors were modeled, lots of experimental data was analyzed, and rules were developed for combining their information. The sonar had the effect of blurring both obstacles and passageways, although was quite accurate in providing distance information to the nearest object. The infrared sensor, a device with stepped power output and varying levels of detector sensitivity, although unable to provide any accurate distance measurement, was able to detect the absence or presence of an obstacle with very good angular resolution. It could very reliably pick out edges of doorways that were invisible to the sonar. However, it didn't work as well as was hoped in finding edges of obstacles when other objects lay just a few feet behind the first. This was due to the fact that the infrared sensor's ranging capability was far too coarse. Rules were developed specifying how sensory data should be combined and the raw data was then refined into a map that was more accurate than if either sensor was used alone.

This refined map was converted into an intermediate representation, the curvature primal sketch, which represented the boundary with knot points that marked significant changes in curvature of the smoothed boundary. The original contour was then approximated by connecting these knot points with straight line segments. From this representation, it was a simple step to convert to a representation expected by path planning programs, namely, a list of polygons. Each straight line segment in the boundary was converted to a list of very thin linked polygons that were translated and rotated appropriately. Examples of such a representation used with a planner based on generalized cones was given.

An even more important reason for choosing the curvature primal sketch as an intermediate representation was that it had certain features which made it amenable to creating a global room map by merging views of the room taken from different perspectives. Because the calculation of the knot points was based on local information, the curvature of neighboring points, they represented landmarks that were invariant from move to move, provided the moves were small. Therefore, these knot points could be used to match and track similarities between data scans taken on subsequent moves. A method for taking several plots and merging them into a global map was discussed. An example of another planner, one based on configuration space, was shown for such a map. Consequently, two examples of intelligent programs, path planners, were shown, which were run not on simulated data, but on real data produced from very cheap sensors.

References

[Bauzil 81]
Bauzil, G., Briot, M. and Ribes, R.
A Navigation Sub-System Using Ultrasonic Sensors for the Mobile Robot HILARE.
In *Proc. First Annual Conference on Robot Vision and Sensory Control.* Stratford-Upon-Avon, UK, 1981.

[Brady 84]
Brady, J. M.
The Curvature Primal Sketch.
Technical Report A.I. Memo 758, MIT, February, 1984.

[Briot 81]
Briot, M., Talou, J. C. and Bauzil, G.
The Multi-Sensors Which Help a Mobile Robot Find Its Place.
Sensor Review :15-19, Jan, 1981.

[Brooks 83]
Brooks, R. A.
Solving the Find-Path Problem by Good Representation of Free Space.
In *IEEE Systems, Man and Cybernetics,* Volume 13. 1983.

[Brooks 84]
Brooks, R. A.
Find-Path For a Puma-Class Robot.
In *International Symposium of Robotics Research.* 1984.

[Brooks 85]
Brooks, R. A. and Lozano-Perez, T.
A Subdivision Algorithm Configuration Space for Findpath with Rotation.
In *IEEE Systems, Man and Cybernetics.* 1985.

[Canny 83]
Canny, J. F.
Finding Edges and Lines in Images.
Master's thesis, MIT, June, 1983.

[Chatila 85]
Chatila, R. and Laumond J. P.
Position Referencing and Consistent World Modeling For Mobile Robots.
In *Proc. IEEE Robotics Conference.* 1985.

[Chattergy 85]
Chattergy, R.
Some Heuristics for the Navigation of a Robot.
Robotics Research, Spring, 1985.

[Coles 69]
Coles, S. L., Raphael, B., Duda, R. O., Rosen, C. A., Garvey, T. D., Yates, R. A. and Munson, J. H.
Application of Intelligent Automata to Reconnaissance.
Technical Report, Stanford Research Institute, Nov, 1969.

[Dobrotin 77]
Dobrotin, B. and Lewis, R.
A Practical Manipulator System.
In *Proc. IJCAI-5*, pages 749-757. 1977.

[Dreyfus 79]
Dreyfus, H. L.
What Computers Can't Do.
Harper and Row, New York, 1979.

[Everett 82a]
Everett, H. R.
A Microprocessor Controlled Autonomous Sentry Robot.
Master's thesis, Naval Postgraduate School, Oct, 1982.

[Everett 82b]
Everett, H. R.
A Computer Controlled Sentry Robot: A Homebuilt Project Report.
Robotics Age, March/April, 1982.

[Everett 85a]
Everett, H. R.
A Second Generation Autonomous Sentry Robot.
Robotics Age, April, 1985.

[Everett 85b]
Everett, H. R.
A Multielement Ultrasonic Ranging Array.
Robotics Age, July, 1985.

[Ferrer 81]
Ferrer, M., Briot, M., and Talou, J. C.
Study of a Video Image Treatment System for the Mobile Robot Hilare.
In *Proc. First International Conference on Robot Vision and Sensory Controls*, pages 59-71. Stratford-Upon-Avon, UK, April, 1981.

[Giralt 77]
Giralt, G., Sobek, R. and Chatila, R.
A Muti-Level Planning and Navigation System for a Mobile Robot; A First Approach to HILARE.
In *Proc. IJCAI-6.* 1977.

[Giralt 83]
Giralt, G., Chatila, R. and Vaisset M.
An Integrated Navigation and Motion Control System for Autonomous Multisensory Mobile Robots.
In *Proc. The First International Symposium of Robotics Research.* 1983.

[Laumond 83]
Laumond, J.
Model Structuring and Concept Recognition: Two Aspects of Learning for a Mobile Robot.
In *Proc. of IJCAI-8.* 1983.

[Lewis 73]
Lewis, R. A. and Bejczy, A. K.
Planning Considerations For A Roving Robot With Arm.
In *Proc. IJCAI-3,* pages 308-315. 1973.

[Lewis 77]
Lewis, R. A. and Johnston, A. R.
A Scanning Laser Rangefinder For A Roving Robot With Arm.
In *Proc. IJCAI-5,* pages 762-768. 1977.

[Lozano-Perez 81]
Lozano-Perez.
Automatic Planning Of Manipulator Transfer Movements.
IEEE Transactions on Systems, Man, and Cybernetics SMC-11(10), 1981.

[Maslin 83]
Maslin, G. D.
A Simple Ultrasonic Ranging System.
Technical Report Polaroid Ultrasonic Ranging System Handbook Application Notes/Technical Papers, Polaroid Corporation, May, 1983.

[Miller 77]
Miller, J. A.
Autonomous Guidance and Control of a Roving Robot.
In *Proc. IJCAI-5.* Cambridge, MA, 1977.

[Moravec 81a]
Moravec, H. P.
Rover Visual Obstacle Avoidance.
In *Proc. IJCAI-7.* IJCAI, 1981.

[Moravec 81b]
Moravec, H. P.
Robot Rover Visual Ncvigation.
UMI Research Press, Ann Arbor, 1981.

[Moravec 83]
Moravec, H. P.
The Stanford Cart and CMU Rover.
Technical Report, Robotics Institute Carnegie-Mellon University, Feb, 1983.

[Moravec 85]
Moravec, H. P. and Elfes, A.
High Resolution Maps from Wide Angle Sonar.
In *Proc. IEEE Robotics Conference.* 1985.

[Nilsson 69a]
Nilsson, N. J.
A Mobile Automaton: An Application Of Artificial Intelligence Techniques.
In *Proc. IJCAI-1.* 1969.

[Nilsson 69b]
Nilsson, N. J. and Rosen, C. A.
Application of Intelligent Automata to Reconnaissance.
Technical Report, Stanford Research Institute, Feb, 1969.

[Polaroid 84]
Commercial Battery Division.
Ultrasonic Ranging System.
Polaroid Corporation, Cambridge, MA, 1984.

[Raphael 68]
Raphael, B.
Programming a Robot.
In *Proc. IFIP Congress 68.* Edinburgh, Scotland, Aug, 1968.

[Rosen 68]
Rosen, C. and Nilsson, N.
Application of Intelligent Automata to Reconnaissance.
Technical Report, Stanford Research Institute, Jan, 1968.

[Thompson 79]
Thompson, A. M.
The Navigation System of the JPL Robot.
In *Proc. IJCAI-5*, pages 335-337. Tokyo, 1979.

[Weiler 77]
Weiler, K. and Atherton, P.
Hidden Surface Removal Using Polygon Area Sorting.
Computer Graphics 11(2), 1977.